Basics of man-machine communication for the design of educational systems : NATO Advanced Study Institute, August 16-26, 1993, Eindhoven, The Netherlands

Institute for Perception Research IPO

Basics of Man-Machine Communication for the Design of Educational Systems

NATO Advanced Study Institute

August 16-26, 1993

Eindhoven, The Netherlands
Organized by IPO

Proceedings

Basics of Man-Machine Communication for the Design of Educational Systems

NATO Advanced Study Institute

August 16-26, 1993

Eindhoven, The Netherlands
Organized by IPO

Organizing Committee

Maddy D. Brouwer-Janse, ASI - Director,
IPO - Institute for Perception Research, Eindhoven, NL

Dominique Béroule, LIMSI-CNRS, Orsay, F
Robbert-Jan Beun, IPO, Eindhoven, NL
Tom Bösser, WWU, Münster, BRD

Editors

Maddy D. Brouwer-Janse, IPO, Eindhoven, NL
Ilse van Kuijck-Aerts, IPO, Eindhoven, NL

Proceedings

Volume I

IPO

9490023

EINDHOVEN

Contents

VOLUME 1

Contents

VOLUME 2

Designing Intelligent Interface for an Advice-Giving System

Ivan Alexandrov, Mariofanna Milanova
Technical University of Sofia
kv. "Strelbishte" bl. 25, ap. 20, Sofia-1404, Bulgaria

1. Introduction

The development of computer techniques nowadays leads to quickly increment of computer users. Most of them are not computer specialists and they don't have time to learn much about computers. That's why the modern computer systems have to take into consideration the features of human mind and to help people to solve their problems using minimum efforts. Creators of computer systems should also take into consideration that users have different levels of knowledges about computers, i.e. the systems can be used by beginners, intermediate specialists and experts.

2. Man-machine communication

Man-machine communication means the interactive exchange of messages between a user and a computer system in order to achive a certain task. A basic principle of man-machine communication consists of user's independent choice of input and the system's totally deterministic reaction. Dialog can be considered as a problem solving method where the user knows the problem and the system is used to solve subproblems [1].

Man-machine communication can be organized in different ways. But they can be classified by 3 levels of abstraction: basic structure, representation and technical realization. Basic structure includes fundamental features which characterize the dialog. These are the problems about the initiator of the dialog, the way the action influence the reaction, the establishment of the task and so on. Representation includes the describtion of messages by which action and reaction are represented (vocabulary, inner and outer format, etc.). Technical realization includes the technical media (input and output facilities) by which the messages are realized.

On the basis of the basic structure 6 types of abstract dialog exist:

- simple question. This is a question by the system with predetermined input interpretation. The user can only enter objects which will be interpreted. This type is mentioned in connection with data collection;
- proposal for selection. The user chooses from a set of alternative tasks proposed by the system. Two variants of this dialog type are menu and yes/no question; - request with syntax for respond. This means a system request on which the user has to react with a syntactically restricted input. The question of the system about the data is a request of such type;
- request for free response. The systems demands from the user a statement in quasinatural language;
- command. The user specifies his tasks and objects according to a prescribed syntax;
- quasi-natural language statement. The user freely chooses a task using a familiar language. This type of dialog imposes at least restrictions on the user.

On the basic of the representation the dialogs can be classified using the following components: vocabulary, inner and outer format of input, formal input redundancy, output syntax and format, semantic properties and others.

3. Describtion of the system

Our team have developed a CAM system which can be connected with any other CAD system (for example, AutoCAD by AutoDESK) and which can generate CNC programs for the following types of detail manufacture: 2D and 3D milling, turning, pocketing, drilling and wire EDM machining [2-3].

The flowchart of the system is given in fig.1.

The interface between the user and the computer system is organized using some menus. At first the user has to select the type of operation he wants to use (milling or turning or pocketing or something else). Then he has to choose the tool and its diameter. The tool can be selected from a tool library where tools are presented as icons. The geometry of the detail has already been given in the CAD system so the user has to tell the CAM system where this geometry is stored. He has also to determine some regimes of manufacture such as spindel, feedrate, etc. After that the system determines the tool path and creates CLDATA file. Using menu, which contains different types of postprocessors, user can select the suitable postprocessor for its CNC machine and as a result CNC program is generated.

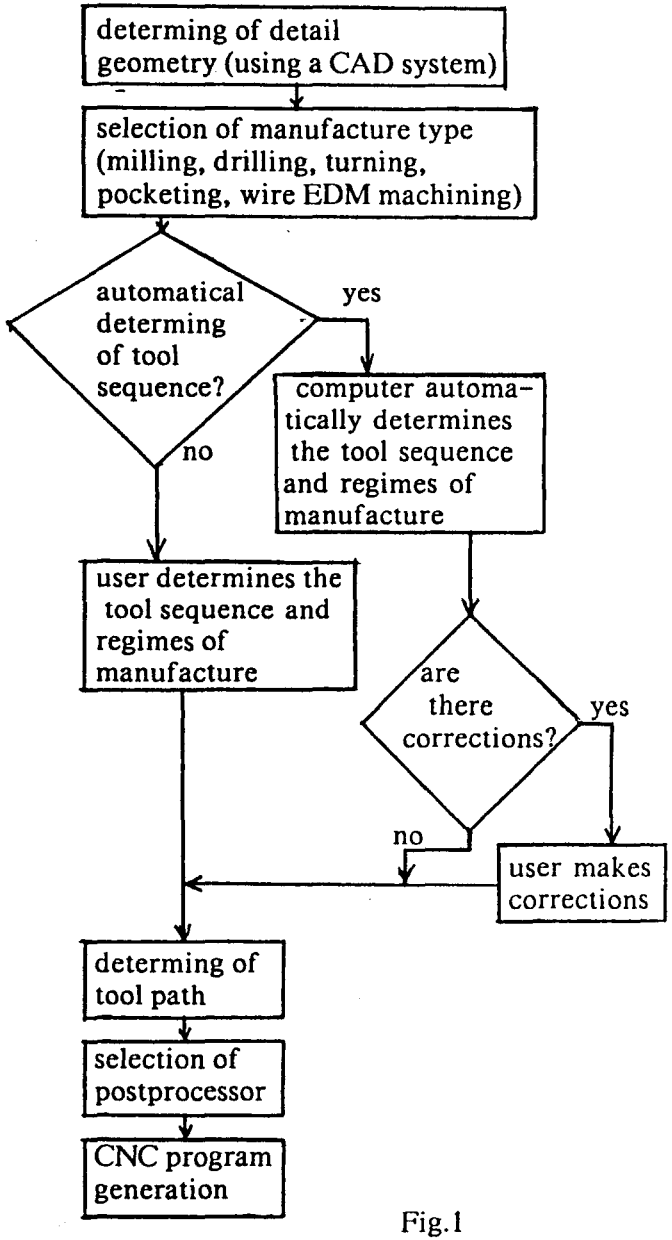


Fig.1

For some types of manufacture an advice-giving module is developed. This module can determine the sequence of tools which can be used for manufacture of a given detail and the regimes of processing. These are the cases of drilling (more precisely, manufacture of holes), milling (more precisely, manufacture of channels) and die blanking in order to reduce its width so it becames approximately equal to the detail width. The sequence of tools and regimes of manufacture given by the advice-giving module can be used directly by users or can be corrected. This module asks user about the sizes and the locations of the holes and channels he wants to manufacture. In order to determine the tool sequence and the regimes of manufacture the system uses algorithms which are developed on the basis of the practice of specialists in this field.

The following types of man-machine communication are used in the system:

- menu. In this case the dialog step consists of the display of section possibilities (action) and the indication of the chosen alternative (reaction). The action has the following basic characteristics: vocabulary (natural language words) and output syntax (quasi-natural language sentences). The reaction has the following characteristics: vocabulary (natural language words), inner format (a sentence out of those displayed in the action) and outer format (free, with an input termination symbol at the end);
- yes/no question. The characteristics of the action are the same as in the previous point (menu). The characteristics of the reaction are: vocabulary ({yes,no}), inner format (a single word is allowed) and outer format (free);
- request with syntax for respond. This type is used when user should determine the sizes of holes and channels;
- command. This type is used when user should determine the location of the holes and channels. He initiates the dialog between him and the computer system when he tells the system that there is a hole or a channel somewhere.

4. Conclusions

A CAM system is developed. This system gives users the possibility to generate CNC programs for detail manufacture. The man-machine communication in this system is developed on the basis of modern techniques. That's why the system is user-friendly so people can work easily with it.

References

1. Dehning W., Essig H., Maass S.: The adaptation of virtual Man-Computer Interfaces to User Requirements in Dialogs, Lecture Notes in Computer Science, 110, 1981.

2. Pavlidis T.: Algorithms for graphics and image processing, Computer Science press, 1982.
3. Spur G., Krause F.: CAD-Technik, Lehr- und Arbeitsbuch fur die Rechnerunterstutzung in Konstruktion und Arbeitsplanung, Carl Hanser Verlag, Wien, 1984

Electronic Books and their Potential for Interactive Learning

Philip Barker

Interactive Systems Research Group, Human-Computer Interaction Laboratory, School of Computing and Mathematics, University of Teesside, Cleveland, UK

Abstract. The growing popularity and importance of electronic books opens up many possibilities for the dissemination of instructional material. Electronic books are made possible because of the availability of low-cost, high-capacity storage facilities based upon the effective use of digital optical storage media - particularly, compact disc read-only-memory (CD-ROM). Because of their portability, robustness and pedagogic potential such books offer a powerful mechanism by which to implement distance learning programmes. This paper describes and discusses the basic nature of electronic books and those special features which make them particularly suitable for the support of both open and distance learning. Two ongoing projects in which we are currently involved are then briefly described.

Keywords. Electronic books, interactive learning, interface design

1.Introduction

Within most modern societies virtually everyone is familiar with books. Indeed, from very early ages (about 12-18 months) young children begin to explore simple picture books thereby learning more about the world they live in and the fantasy worlds that we create for them. Subsequently, they learn to select the particular books that they wish to look at and acquire the skills needed to turn the pages of a book and identify the objects that are portrayed upon these pages. Later, as they grow older and their skills develop further, children 'move' from picture books to illustrated story books and subsequently to novels and other forms of the 'printed word'.

From what has been said above it is easy to see that books form an important part of human culture. They are used to support a wide range of human activity - such as education, entertainment, business and research. Indeed, for many professions (such as law, engineering and science) books form the basic mechanism by which information is stored and made available to others who may wish to use it.

In terms of their functionality, the purpose of books is thus twofold. First, they can act as repositories for information which we wish to keep - possibly, for ever. Second, they can act as a mechanism by which one person (or a group of people) - the author(s) - can communicate with another group of people (the readers). Obviously, as was

hinted above, the material that is stored and/or communicated within the pages of a book may be fact, fiction or fantasy. Over the centuries, since the introduction of the printing press, conventional books have obviously played a fundamental role in a wide range of information dissemination and knowledge transfer activities. Some years ago we analysed the role of conventional books in technical knowledge dissemination processes [1]. We identified a number of important limitations of such books (see table 1) and suggested that some new form of book was needed in order to overcome the limitations of those that are printed on paper.

Table 1 Basic Limitations of Conventional Books

- difficult to reproduce
- expensive to disseminate
- difficult to update
- single copies cannot easily be shared
- easily damaged and vandalised
- bulky to transport
- embedded material is unreactive and static
- cannot utilise sound
- cannot utilise animation or moving pictures
- unable to monitor reader's activity
- cannot assess reader's understanding
- unable to adapt material dynamically

We used the term 'electronic book' to describe a new form of book whose pages were composed, not of static printer's ink, but from dynamic electronic information. Generally, we now use this new term to describe information delivery systems that are capable of providing their users with access to pages of reactive electronic information with which they can interact. As we shall discuss later, the pages of information which make up an electronic book are organised conceptually just like the pages of a conventional book.

Obviously, the properties of a book will depend critically upon the medium upon which it is published. Nowadays, three basic media are commonly used for this purpose: paper, magnetic disc and optical disc. Unfortunately, publication on paper renders the information embedded in a book static and unreactive. In order to make its information reactive and dynamic the book concerned must be published on a suitable 'interactive medium' [2]. Most electronic books will therefore be published on either magnetic or optical storage media. Although books published on magnetic media are useful, they are not as useful as those that are produced using digital optical storage media. The substantial utility of optical storage media arises from its large capacity, robustness, and relative cheapness. Two types of optical storage media are commonly used for producing electronic books: magneto-optical re-writable disc storage [3] and compact disc read-only-memory (CD-ROM) [4].

Although a considerable amount of work is undertaken using re-writable optical disc (mainly for testing and prototyping purposes), the optical medium that is most often used for electronic book publication is CD-ROM. As well as its durability and stability, the other attractive feature of CD-ROM as a publication medium for electronic books is its immense storage capacity. A single CD-ROM is able to store a total of 650 Mbytes of information. In terms of the magnetic storage available of a low-capacity 5.25 inch floppy disc, this is equivalent to about 1800 floppy discs!

The 650 Mbytes of storage available with a CD-ROM disc can be used in a variety of different ways for the storage of multimedia information such as text, sound, static pictures, animation, computer programs and a very limited amount of motion video. Typical figures that are often quoted to reflect the storage capacity of a CD-ROM disc are: 200,000 pages of A4 text; or 20,000 low-quality (PCX) image files; or 2,000 TV quality still images; or 30 seconds of video; or 18 hours of low-quality sound. The way in which the available storage is used within a given electronic book production will depend critically on the 'media mix' needed by the particular publication concerned.

Obviously, the actual amount of material that can be stored on a CD-ROM disc will depend upon whether or not any form of data compression technique is applied to the information before it is committed to storage. Normally, in order to store full-motion video pictures on a CD-ROM various types of compression (and decompression) techniques must be applied.

From the point of view of electronic book publication, the most commonly used CD-ROM format is that specified in the ISO 9660 standard. This is an internationally recognised way of storing information on a CD-ROM so that it can be read by any computer system that has a suitable disc drive attached to it. Unfortunately, the ISO 9660 standard does impose some restrictions on what can be done with a CD-ROM disc. In order to overcome these restrictions, several other approaches to using CD-ROM are rapidly emerging. The four most important of these are: CD-ROM XA (extended arhitecture); CD-I (compact disc interactive); CDTV (Commodore's Dynamic Total Vision); and DVI (Digital Video Interactive). Each of these offers many exciting possibilities for the design of electronic books.

For a variety of reasons, electronic books are rapidly becoming an important pedagogic resource. Because of their importance in the context of learning and training activity, the remainder of this paper discusses a number of important issues relating to their production and use. We shall first discuss their basic architecture and some of the important factors that need to be considered during their design and fabrication. A taxonomic framework (which enables books of this sort to be classified) will then be presented. Finally, an outline will be given of the nature of telemedia books and how they can be used to support distance learning.

2 Basic Architecture

Unlike conventional books, those that are published on electronic media require some form of 'delivery platform' to facilitate access to them. Unfortunately, although a large number of electronic books exist, there seems to be no 'standard' delivery

platform that will enable all the currently available electronic books to be accessed. Indeed, each particular type of electronic book that is presently available seems to need its own specific tailor-made delivery platform. Three examples will be used to illustrate this point.

Consider first the 'expanded books' produced by the Voyager Company for delivery platforms based upon suitably configured Apple Macintosh computers. 'Jurassic Park' [5] is just one example of the many electronic book titles that have recently been released by this company. Each one consists of a single 3.5 inch magnetic disc containing compressed multimedia information that must be loaded onto a hard disc before it can be accessed. Once the contents of the disc have been installed the electronic text and diagrams provide a 'dynamic medium' that enables a user to 'become a more active reader'.

As a second example consider the electronic books that have been produced for delivery on the Sony Data Discman electronic book player [6]. Hutchinson's 'Guide to the World' [7] is a typical example of a publication that has been produced for delivery using this platform. Each of the available publications consists of a single 8 cm digital optical storage disc (of the read-only variety) embedded within a caddy that makes its appearance very similar to the expanded books that were described above. Although similar in appearance, these books are totally incompatible with those intended for the Macintosh or any other currently available delivery platform.

The third example is taken from an electronic book series being produced by the Elsevier publishing company - known as the 'Active Library' series. A typical example of a publication taken from this series of books is the 'Active Library on Corrosion' [8]. Each of these books is published on a 12 cm CD-ROM disc for delivery using an IBM PC platform that is equipped with the Microsoft Windows 3.1 graphical user interface. Again, this electronic publication (and its host delivery platform) is totally incompatible with either of the other two products that have been previously mentioned.

The above three examples are just a few of the many others that could be cited. Each of the electronic books is excellent in its own right but, unfortunately, each one requires a different kind of delivery platform. There is therefore no compatibility between products.

Despite the gloomy incompatibility problems outlined above some rationale is starting to emerge with respect to electronic book architectures. This architectural rationale is emerging along two different basic directions. First, in terms of the fundamental structure and composition of delivery platforms; and second, in terms of how users are meant to perceive the conceptual structure of an electronic book publication.

The basic architecture and composition of a typical electronic book delivery platform is illustrated schematically in figure 1. Four basic components are needed. First, either an embedded or an explicit computer facility (this is required for control purposes and in order to achieve overall system integration). Second, a multimedia information storage device (such as a CD-ROM unit). Third, a display facility that enables users to see (and/or hear) the information contained in an electronic book.

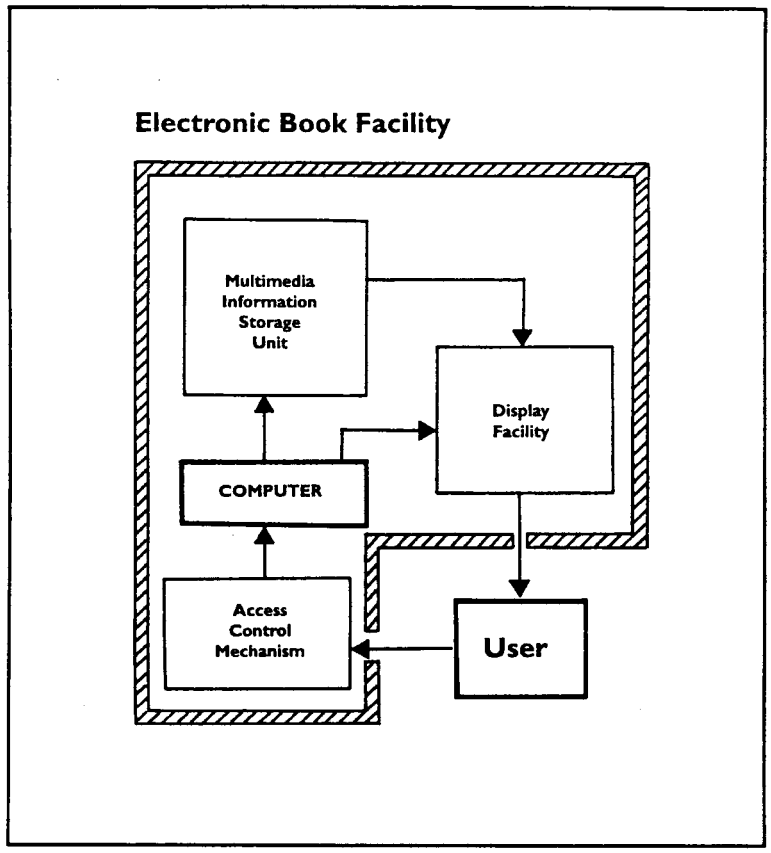


Fig. 1. Basic structure of an electronic book delivery station

Fourth, a suitable access control mechanism. This latter component will usually consist of a suitably designed hardware interface (such as a keyboard of a mouse) and a software environment to control the retrieval and presentation of information. The nature of electronic book software varies considerably and can be used in many different ways to influence how users will perceive a particular publication.

One useful conceptual model that now underlies the development of many electronic book publications is illustrated schematically in figure 2. This depicts an electronic book in terms of a collection of reactive and dynamic pages of multimedia information (that can embed text, pictures and sound). The information contained within these pages is of three basic types: aesthetic (which is used both to help reinforce the underlying book metaphor and also to provide an ergonomically 'pleasing' appearance); informative (which is intended to instruct or inform those who use a particular book); and either implicit or explicit control functions. The control options that are available are important because they enable users to specify the nature of the information that they wish to retrieve from a given book and how this retrieved material is to be displayed within the confines of the host delivery

platform. Some examples of simple primitive control options for electronic books are illustrated in figure 2. The most commonly used functions are: next page; previous page; goto page N; exit book; and so on.

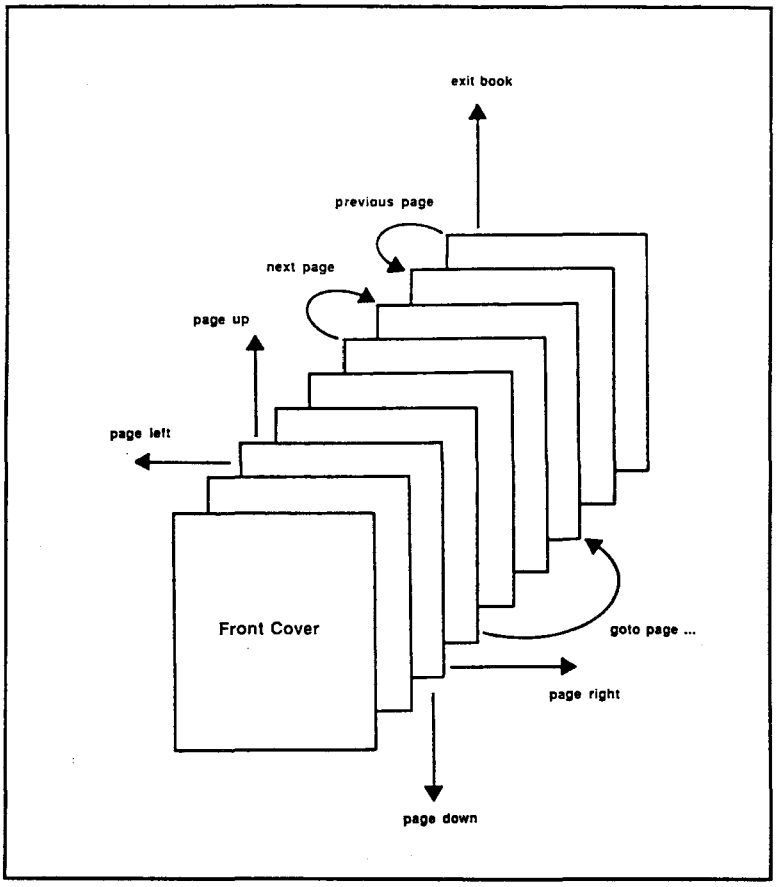


Fig. 2. Conceptual model for an electronic book

The basic control options that are used within the pages of an electronic book may take the form of icons, dynamic menus, dialogue boxes, scroll bars and/or hotspots. The reactive areas that make up hotspots may be embedded within chunks of screen-based text and/or pictures (these form the basis of the 'hypermedia' electronic books that we discuss later). Obviously, the particular combination of control options made available within any given publication will depend on a variety of factors - such as: the type of book involved; the purpose for which it is intended; the logical and physical sizes of its pages; whether it has a linear or non-linear structure; and so on.

When designing and producing an electronic book it is important to give ample consideration to all those design factors that are likely to influence the overall impact, effectiveness and efficiency of the final publication. Some of the factors that we have found to be important are discussed in the following section.

3 Electronic Book Production

The design, fabrication and dissemination of electronic books is rapidly becoming a major area of activity both within education and within the publishing industry. Because of the importance of this topic our 'electronic book' project (which commenced in 1990) [9] was intended to explore the use of CD-ROM for the publication of electronic books. Two particular objectives that we had in mind were: (a) to try and formulate a set of design guidelines to facilitate electronic book production (paying particular attention to the role of end-user interfaces); and (b) to assess the potential of electronic books as a mechanism for the distribution of interactive training and learning resources for use in distance education and flexible learning environments. This section of the paper concentrates on the first of these objectives; the second objective is described in more detail later.

During our project two basic research and development strands were pursued. The first of these involved an evaluative study of a range of commercially available electronic book publications (such as the Grolier Encyclopedia and Compton's Multimedia Encyclopedia). The second strand involved the design, production and controlled evaluation of a number of in-house productions. Three different in-house electronic book demonstrators were produced [10]. These provided examples of: (1) a hypermedia book (entitled 'An Electronic Book for Early Learners'); (2) a multimedia book (entitled 'Screen Design for Computer-based Training'); and (3) an intelligent electronic book (called 'a Static Picture Book with Audio Narrations'). Each demonstrator was designed to explore a different aspect of electronic book production.

In our research project we discovered that two basic types of design tool were needed to facilitate the creation of electronic books. The first of these was a set of high-level design models while the second constitutes a collection of more pragmatic low-level 'tips' (or guidelines) relating to: (1) end-user interface design; (2) the way information is organised on CD-ROM; and (3) the effective creation of access stations for use with electronic books.

Undoubtedly, the most important of our findings was the set of three high-level development models that we formulated. We refer to these as the 'conceptual' model (shown in figure 2), the 'design' model and the 'fabrication' model. The first of these is intended for end-users of electronic books (as an orientation tool). The second and third models are intended for designers and producers of electronic books as they describe architectural and procedural issues, respectively.

When designing electronic books we found that the basic design model shown in figure 3 was an extremely useful asset [10-12]. A major objective of the initial design phase of an electronic book is the formulation of the end-user interfaces that will be used to enable users to access the information that is held within it [13]. Book and page structures then have to be decided upon [14]. The content of the book must then be specified. Finally, the nature of the 'reader services' (browsers, bookmarks, glossaries, etc) must be agreed upon. As can be seen from figure 3, the use of a suitably designed knowledge corpus is fundamental to the creation of our electronic books.

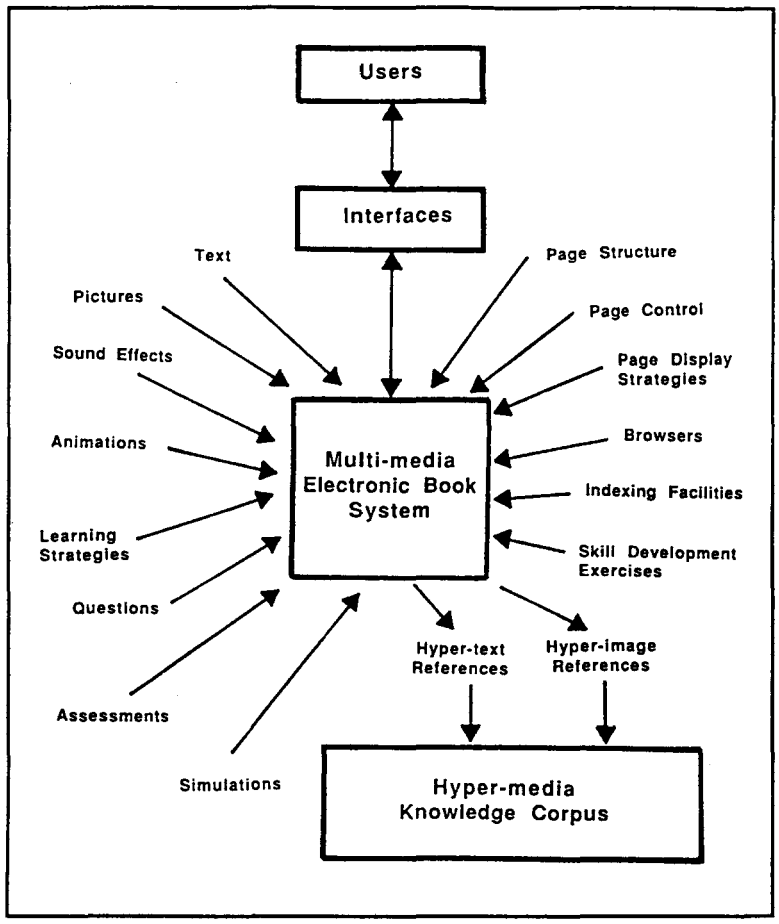


Fig. 3. Development model for electronic books

Obviously, the attractive feature of CD-ROM as a publication medium is the large capacity that it offers for the storage of multimedia and hypermedia knowledge corpora.

The creation of electronic books requires a number of different development stages. These need to be rigorously adhered to if efficient production is to be achieved. The fabrication model is therefore used to describe the relationships between the various stages that are involved in transferring materials from the initial development phase (based upon the use of hard disc and re-writable optical disc for CD-ROM emulation) through prototyping through to the final production stage (using read-only discs). Obviously, it is these latter discs (in ISO 9660 format) that are distributed to users of our products. The fabrication model that we use to support our electronic book productions is illustrated schematically in figure 4.

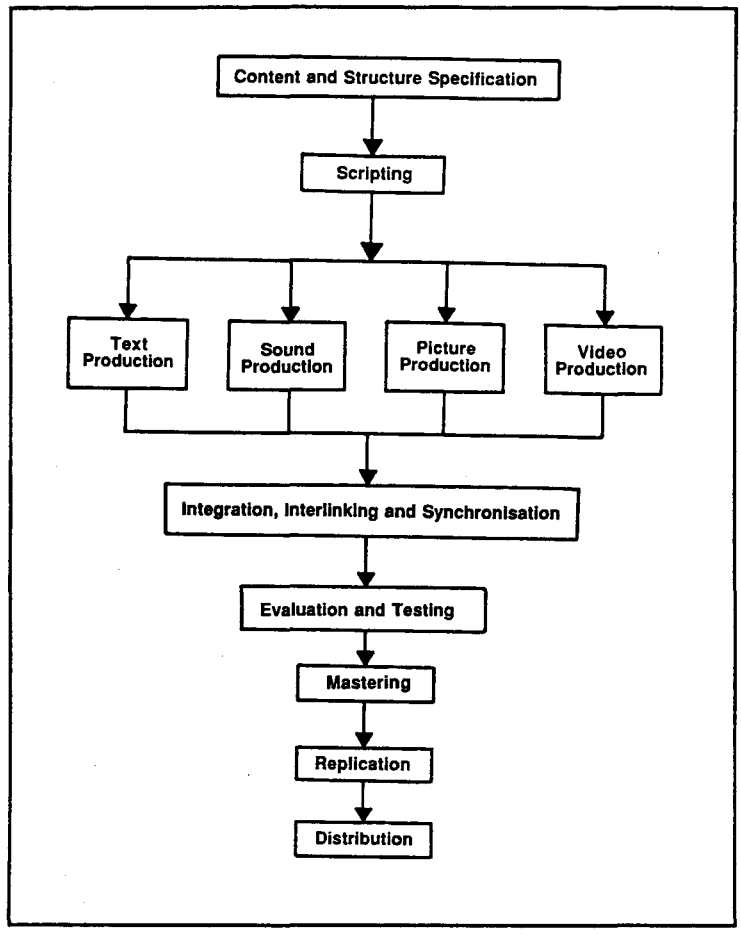


Fig. 4. Fabrication model for electronic books

As can be seen from this figure the initial development phase of an electronic book project involves specifying the content and structure of the book. These are then brought together by means of the scripting process. The script gives a detailed specification of the material that is to be embedded within the book (on a page by page basis) and the relationship between the pages, control mechanisms and end-user tools that are to be made available. It will also embed details of all the textual, sonic and pictorial information that is needed for the electronic book pages. The creation of the script will therefore involve the simultaneous production of all the multimedia resources needed for the book. Once these have been created and individually tested the next phase of electronic book production can commence. This will involve integrating and synchronising the multimedia resources and (if a hypermedia book is being produced) interlinking them in appropriate ways.

When the integrating, interlinking and synchronising phase of electronic book production is complete the evaluation and testing phases can commence. This may

involve the generation of various kinds of 'prototype book'. Once the results of the evaluation and testing have been obtained any final amendations can be incorporated before the final CD-ROM product is mastered, replicated and distributed to users.

During the course of our research into electronic book production we were able to formulate a range of useful design guidelines. We took some trouble to document these as we anticipated that they would be of help to other people who might wish to become involved in electronic book production. The series of guidelines that we produced fell naturally into the following six basic categories: knowledge engineering; page design; interaction styles; end-user tools and services; use of multimedia; and use of hypermedia. Further detailed descriptions of the actual guidelines themselves are given elsewhere [10,13,15].

4 Types of Electronic Book

Electronic books can be classified in a variety of different ways depending upon the medium that they are published on, the functions they perform and the types of facilities and services that they provide. One very simple taxonomy proposed by Barker and Giller [16] categorises electronic books into four basic classes: (1) archival; (2) informational; (3) instructional; and (4) interrogational.

The first category of book offers a method of storing large volumes of information relating to some particular subject area. Within such books the end-user interface will normally be designed in such a way that it will permit a variety of different methods of information retrieval. Examples of such books include large catalogue systems and databases of records and data. The Grolier Encyclopedia [17] and Compton's Multimedia Encyclopedia [18] are two such examples of this category. In many ways, electronic books that fall into the informational category overlap with those in category one. However, the stored information is usually less comprehensive and more specific - relating to a particular topic area. An example of this category of electronic book is the Oxford Textbook of Medicine on Compact Disc [19]. The third category of electronic book listed above (instructional) is intended to provide a means of achieving highly efficient and effective skill and knowledge transfer mechanisms for the support of learning and training activities. Users of such books are given the opportunity to learn and progress at their own pace using their own particular style of learning. Some electronic books in this category will actually assess and adapt to its user's personal learning style. Such books automatically re-configure the material that is presented so as to accommodate its user's preferred approach to learning. Our 'Screen Design for Computer-based Training' [10] is an example of an instructional electronic book. The intention of the last category of electronic book (interrogational) is to support testing, quizzing and assessment activities which will enable readers to gauge their depth of knowledge about a particular topic. This type of book contains three essential components: a question (or exercise) bank; a testing and assessment package; and an expert system. The latter is used to analyse a reader's responses and deduce an appropriate grade or level of competence based on these responses.

Although the taxonomy described above is a useful one, it is often advantageous to use one which is more 'fine-grained'. In view of this requirement, we now propose a taxonomy that contains ten basic classes of electronic book [2,14]. Depending upon the type of information that they embed and the kinds of facility that they make available, we now suggest that electronic books can be classified into the following basic categories: text books; static picture books; moving picture books; talking books; multimedia books; polymedia books; hypermedia books; intelligent electronic books; telemedia electronic books; and cyberspace books. Each of these categories of electronic book is briefly discussed below.

As their name suggests, text books are composed of pages of textual material that have been organised into suitably sized 'chunks' of information. The chunk size that is employed will depend upon the screen size that is used for information display and the number of chunks/page that it is required to present simultaneously. Static picture books consist of a collection of pictures that are organised into some particular theme; the pictures may be of various 'qualities' with respect to their resolution and range of colours that they embed. Moving picture books are constructed from either animation clips or motion video segments - or combinations of each of these; the 'mix' used will depend upon a variety of factors such as the purpose of the electronic book and the 'message' that it is to 'convey'. Talking books depend for their success upon recorded sound (both high- and low-quality) that is used in conjunction with a variety of 'interactive audio' techniques to facilitate end-user control of information and knowledge transfer.

Multimedia books use various combinations of two or more communication channels (either in sequence or simultaneously) in order to encode a particular message. Such books use text, sound, pictures and moving images that are basically organised in a 'linear' fashion. The materials are delivered by means of a single delivery medium (such as magnetic disc or CD-ROM). Polymedia books, in contrast to multimedia books, use a combination of several different media (CD-ROM, magnetic disc, paper, and so on) in order to deliver their information to end-users. Hypermedia electronic books have much in common with multimedia books in that they depend upon the use of multiple communication channels. However, unlike multimedia books, hypermedia books employ 'non-linear' organisations of information based upon the use of web-like structures [20]. Because of the embedded intelligence that they contain, intelligent books are in many ways similar to the 'interrogational' books described by Barker and Giller [16]. These books are capable of dynamic adaptation as a consequence of interaction with end-users. Undoubtedly, two of the most exciting types of book that we are currently developing are telemedia books and cyberspace books. The first of these uses telecommunication facilities to augment the capabilities of a CD-ROM publication in order to support highly interactive distributed distance learning activities [21-23]. Cyberspace books are used as a means of providing their readers with access to various types of virtual reality facility; such books employ different kinds of interactive simulation environment in order to provide end-users with participative, 'real-life' encounters that they would not normally be able to experience.

Because of their importance in the context of distance and independent learning activities, telemedia books will be discussed in more detail in the following section of the paper.

5 Telemedia Books and Distance Learning

As we have suggested above, electronic books can be used to support a wide range of learning and training applications. They are particulary important in the context of supporting distance learning, individualised self-supported learning and co-operative group learning at a distance [21,23]. The way in which we envisage this happening is illustrated conceptually in figure 5.

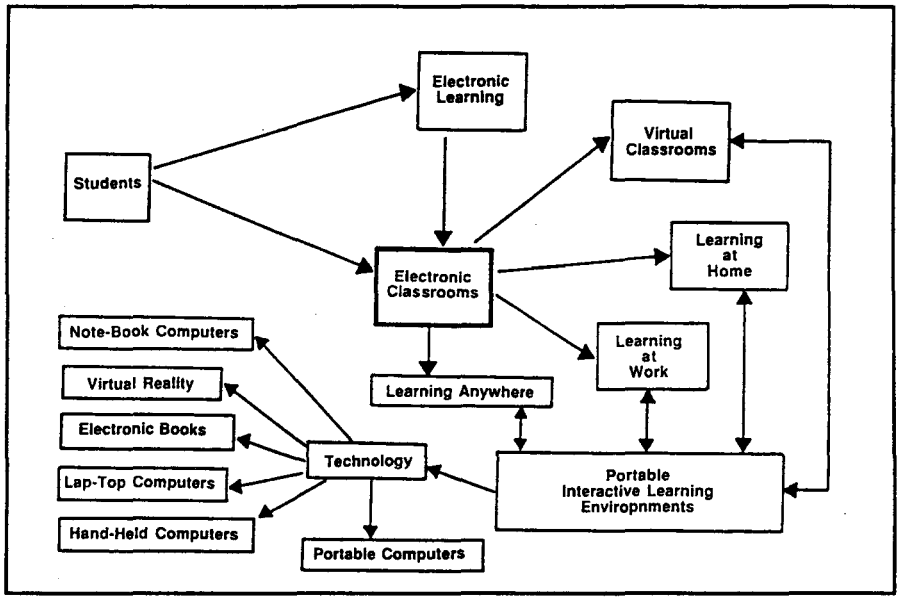


Fig. 5. The role of electronic books in distance education

Embedded within this figure is the idea of students learning by electronic means through the use of electronic classrooms. Such classrooms may exist in two basic forms. First, they may exist physically in a particular geographical location - being composed of a relatively small number of interactive workstations contained within a given room or building. Second, they may exist in the form of a 'virtual classroom' that is composed of an almost limitless number of learning stations that are physically distributed anywhere in the world - for example, in home environments, in people's places of work, in public places such as libraries, and so on.

We envisage that a very large proportion of the delivery platforms for interactive learning will be of a highly portable nature - based upon the use of various types of portable computer system (such as lap-tops, hand-held computers, notebook computers and various sorts of consumer products based upon technologies such as CD-I and CDTV). We refer to such environments as 'portable interactive learning environments' [22]. The use of such technology is attractive because it means that many aspects of learning and training can transcend institutional boundaries. Of course, it is anticipated that all the workstations used to support this type of learning will incorporate the type of architecture that was previously illustrated in figure 1. They will therefore be capable of making large amounts of multimedia and/or hypermedia information available through the medium of electronic books that are published on optical media such as CD-ROM.

Obviously, the use of stand-alone workstations similar to those described above (and illustrated schematically in figures 1 and 5) would only go part of the way to realising the overall needs of learners and trainees in a virtual classroom situation. It is therefore important to consider what other functions a portable learning environment needs to provide. Naturally, an important aspect of conventional classrooms that must be provided within virtual classrooms is 'class contact' with other students and tutors. In order to meet this requirement it is necessary to provide various forms of 'person to person' communication facilities (for example, by means of electronic mail, bulletin boards, telephone, video-phone and/or conferencing facilities). Such facilities not only allow fellow students to communicate with each other but also support, if necessary, communication with tutors and subject matter experts who make themselves available for use as 'human learning resources'. The type of computer-based learning environments needed to support this approach to learning and training is illustrated schematically in figure 6.

Fundamental to this type of workstation is the presence of an appropriate connection to a host telecommunication facility. This may take a variety of different forms. It might be a simple modem and 'dial up' connection (made through a telephone network) to some other compatible modem attached to a remote host computer. Alternatively, the workstation may be connected directly (via a suitable network card) to a local area network and then, through a series of wide area networks to some remote site that might be located almost anywhere in the world. Obviously, as well as supporting distant person-to-person communication such workstations are also able to support access to a wide range of other remote resources such as electronic libraries and either down-loadable or shared cyberspaces.

As we have suggested earlier, electronic books that are designed in such a way that they are able to take advantage of a telecommunications infra-structure similar to that described above are referred to as telemedia books. We envisage a variety of different uses for such books within the context of distributed distance learning both within academic and non-academic organisations. Two examples of systems that we are currently developing will be briefly described in order to illustrate how such books may be used.

The simplest type of telemedia book for use in a delivery platform similar to that shown in figure 6 is one which uses CD-ROM for the bulk publication of large

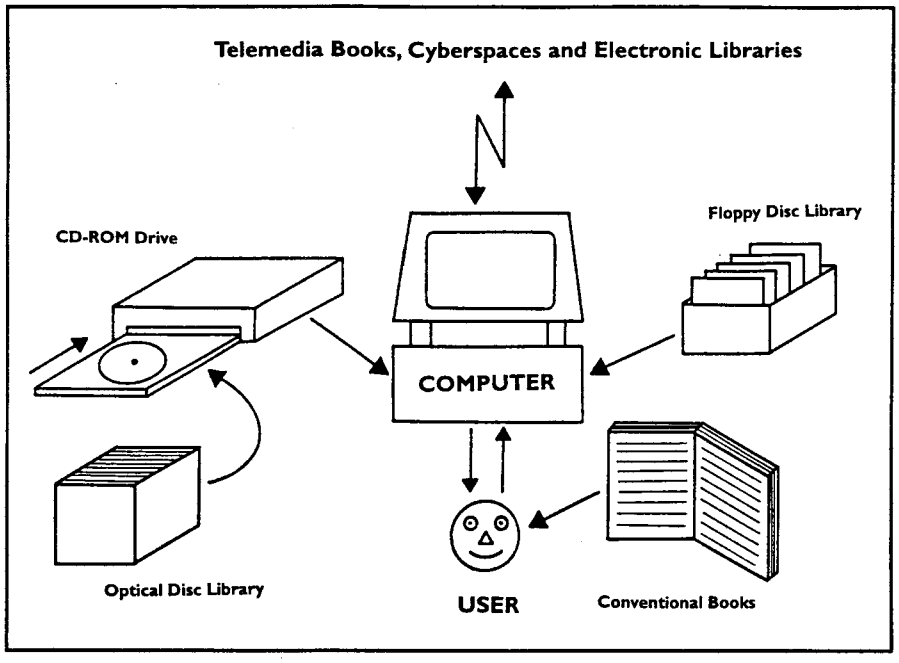


Fig. 6. Polyfunctional workstation for electronic book delivery

amounts of interactive course material for use by students. However, also embedded in the electronic book (in the form of a reader service) is an electronic mail facility that is referenced through an icon. This facility can be used in a variety of ways to support communicative exchanges between students and for the transfer and/or sharing of materials. The work that we are currently undertaking in our 'interactive language learning project' illustrates this approach to the use of telemedia books for learning French [24]. The essential course material is embedded upon a CD-ROM (as an electronic book) that allows individual students to listen to and practice speaking French. They can also test their writing ability by sending electronic mail communications to each other and to fellow students and tutors located both within the UK and in France.

The second telemedia book project that we are currently working on involves the design of support material in the form of 'interactive manuals' for use within an electronic performance support system (EPSS) that is to be used within the context of office automation [25]. The EPSS system is intended to provide just-in-time (JIT) training at a particular point of need. The system that we have been designing and prototyping is intended to offer this type of support within a geographically distributed multi-centre organisation. The interactive manuals that embed the technical and procedural information necessary for the organisation are published on CD-ROM. The communications infra-structure running above this basic publication level is accessed as a standard facility within a telemedia 'company services' handbook that enables employees to gain access to each other and to sources of expert

help and advice. This enables employees to share skills and when necessary develop new ones through tele-tutoring techniques and JIT methods.

6 Conclusion

Books are an important mechanism for the storage and communication of information. Conventional books are published on paper as a collection of pages of static information. Therefore, in many ways, the concept of a book is intimately bound to this medium. This has both advantages and disadvantages - some of which have been discussed in this paper. Of course, there is no inherent reason why the concept of a book has to be media dependent. Other media could equally well be used to publish books or embed the book concept. For example, conventional books could be 'televised' on television but because of the low interactivity of this medium users would loose the ability to 'turn pages'. However, by using a more interactive medium (such as a computer) to publish a book, many of the properties of conventional books can be emulated. It is therefore important to realise that the properties of a book will depend very much upon the nature of the medium (or 'media mix') used for its publication. In this paper we advocate the use of telemedia electronic books as a useful resource for promoting and supporting distance education and cooperative group learning at a distance. Two examples of the use of electronic books for distance learning have been briefly described. Obviously, many more exciting possibilities of this approach yet remain to be explored.

7 References

1. Barker, P.G. and Manji, K.A., New Books for Old, Programmed Learning and Educational Technology, 25(4), 310-313, (1988).
2. Barker, P.G., Interactive Electronic Books, Interactive Multimedia, 2(1), 11-28, (1991).
3. Paris-Roth, J., Re-writable Optical Storage Technology, Meckler, Westport, CT 06880, USA, 1991.
4. Lambert, S. and Ropiequet, S., CD-ROM: The New Papyrus - The Current and Future State of the Art, Microsoft Press, Redmond, WA 98073-9717, USA, 1986.
5. Crichton, M., 'Jurassic Park', The Voyager Company, Santa Monica, CA, USA, 1991.
6. Rockman, S., Sony Data Discman DD-1EX, Personal Computer World, 15(3), 262-266, (1992).
7. Hutchinson's Guide to the World, Electronic Book Version 1.0, Helicon Publishing, London, UK, 1992.
8. Bogaerts, W.F. and Agema, K.S., Active Library on Corrosion, Elsevier Science Publishers, Amsterdam, The Netherlands, 1992.
9. Barker, P.G. and Giller, S., Design Guidelines for Electronic Book Production - Overview Report, Interactive Systems Research Group, School of Computing and Mathematics, University of Teesside, Cleveland, UK, 1992.

10. Giller, S., Design Guidelines for Electronic Book Production, MPhil Thesis, University of Teesside, Cleveland, UK, 1992.
11. Barker, P.G., Electronic Books, Learning Resources Journal, 6(3), 62-68, (1990).
12. Barker, P.G. and Manji, K.A., Designing Electronic Books, Journal of Artificial Intelligence in Education, 1(2), 31-42, (1990).
13. Richards, S., End-User Interfaces to Electronic Books, Draft PhD Dissertation, Interactive Systems Research Group, School of Computing and Mathematics, University of Teesside, Cleveland, UK, 1992.
14. Barker, P.G., Electronic Books, Special Edition of Educational and Training Technology International, 28(4), 269-368, (1991).
15. Barker, P.G. and Giller, S.A., Design Guidelines for Electronic Book Production, Final Project Report submitted to the Learning Technology Unit, Training, Enterprise and Education Directorate, Department of Employment, Moorfoot, Sheffield, 1992.
16. Barker, P.G. and Giller, S., Electronic Books, 179-184 in Aspects of Educational Technology, Volume XXV: Developing and Measuring Competence, edited by D. Saunders and P. Race, Kogan Page, London, 1992.
17. Grolier Inc., The New Grolier Encyclopedia - User's Guide, Grolier Electronic Publishing Inc, Sherman Turnpike, Danbury, CT, USA, 1988.
18. Britannica Software, Compton's Multimedia Encyclopedia, Britannica Software Inc., San Francisco, California, USA, 1990.
19. Weatherall, D.J., Ledingham, J.G.G. and Warrell, D.A., The Oxford Textbook of Medicine on Compact Disc, Second Edition, Oxford University Press, Oxford, 1989.
20. Barker, P.G., Hypermedia Electronic Books, in Proceedings of the Seventh Canadian Symposium on Instructional Technology, Montreal, Quebec, 6th-9th May, 1992.
21. Barker, P.G. and Giller, S., Electronic Books for Distance Learning, 759-761 in Volume 2 of the Proceedings of the Ninth International Conference on Technology and Education, Paris, France, 16-20 March, 1992.
22. Barker, P.G., Portable Interactive Learning Environments, paper presented at the AETT '92 International Conference, University of York, 6th-9th April, 1992.
23. Barker, P.G., Electronic Books and their Potential for International Distance Learning, Invited paper presented at the East-West Conference on 'Emerging Computer Technologies in Education', Moscow, 6-9 April, 1992.
24. Interactive Language Learning Project, National Council for Educational Technology, Sir William Lyons Road, Science Park, University of Warwick, Coventry, CV4 7EZ, 1993.
25. Banerji, A.K., Designing Electronic Performance Support Systems, Draft PhD Thesis, Interactive Systems Research Group, School of Computing and Mathematics, University of Teesside, Cleveland, UK, 1993.

Summary NATO Summerschool

Robbert-Jan Beun

Institute for Perception Research/IPO
P.O. Box 513
5600 MB Eindhoven, The Netherlands

A framework for speech-act generation in cooperative dialogue

A central theme in Artificial Intelligence approaches to speech acts is the modelling of the cognitive-state changes of the participants in terms of beliefs and intentions (e.g., Perrault 1990). Although this so called *context-change* approach seems to offer an attractive formal treatment of cognitive-state changes of dialogue participants, the approach lacks the power to model the generation of speech acts in a conversation. Generation of a speech act is of crucial importance to determine, for instance, adequate behaviour of an information exchange system in response to a user's primary communicative goals.

To determine adequate communicative behaviour of participants (human or machine) in a cooperative dialogue, a theoretical framework will be presented that is based on the concepts of *dialogue game* (Carlson 1985), *preference organization* (Levinson 1983) and *cognitive state(-change)* of the participants. Main goal of the dialogue is to exchange information in a collaborative manner so that, if the information is available, in the final state the participants mutually believe the answer to an initial question. The participants' communicative strategy is determined by the rules of the dialogue game. Three types of rules will be considered: a. preference rules that tell us which speech act is preferred in a certain state, b. prohibitory rules that forbid the performance of certain speech acts and c. closing rules that define the criteria for ending the conversation. The resulting structure of the information exchange can be complex, since the knowledge to find the answer to the initial question may be distributed among the participants and may therefore result in subdialogues and the generation of counter-questions.

By means of simple examples, it will be shown that the application of different communicative strategies and different initial belief states give important theoretical insight in the conversational structure of the dialogue. Moreover, it is claimed that, if the type of dialogue is well-restricted, only three types of belief are significant to determine adequate behaviour of an information exchange system in a cooperative situation: a. the system's belief about the domain of discourse, b. the system's belief about the mutual belief of the system and user and c. the system's belief about what the user does not believe.

References:

Carlson, L. (1985) *Dialogue Games*. Dordrecht: Reidel Publishing Company.

Levinson, S. (1983) *Pragmatics*. Cambridge: Cambridge University Press.

Perrault, C.R. (1990) An application of default logic to speech act theory. In: P.R. Cohen, J. Morgan & M.E. Pollack (Eds.), *Intentions in Communication*. Cambridge, Mass.: MIT Press.

Cooperative Problem Solving as a Basis for Computer Assisted Learning

Sviatoslav Brainov

Institute of Mathematics, Sofia, Bulgaria

Abstract. In this paper we present a new approach for computer assisted learning. This approach incorporates features of the intelligent tutoring systems and distributed problem solving. It has been designed to permit the multiperson - multimachine interaction and to stimulate the social activity of the students. A framework called DEE (Distributed Educational Environment) that supports multiagent learning activity on the basis of cooperative problem solving is described. The DEE is organized as a network of autonomous problem solvers.

Keywords. Computer assisted learning, distributed problem solving, cooperation, task allocation, conflict resolution, blackboard-based systems.

1. Introduction

The proliferation of the computer networks and the recognition that much human activity involves groups of people have provoked the interest in distributed problem solving. One of the main approaches in the distributed problem solving is the person-machine coordination. This approach provides useful means for managing a collection of people and machines working together in coordinated and cooperative way. Within this approach research in distributed problem solving has great impact in the area of computer assisted learning. Currently accepted paradigm in the computer assisted learning of one (student)-to-one(computer) interaction may be improved by absorbing some ideas from multiagent interaction theory.

In this paper we present a framework called DEE (Distributed Educational Environment) that supports multiagent learning activity on the basis of cooperative problem solving. The DEE is organized as a network of autonomous problem solvers. Some of the problem solvers are extended with teaching abilities. Students may participate in the network by one of the existing nodes. The DEE may be considered as a departure from the conventional one-to-one teaching model to one that engages many learners

and teachers. In our opinion the process of the learning as a coordinated multiagent activity more directly reflects the nature of human learning.

The remainder of this paper commences with the motivations behind the our approach. We then describe the architecture of the DEE and organization of communications. Finally, we summarize our approach and propose some open problems.

2. Motivation

One of the major difficulties which arises when using a conventional educational system is connected with resource limitations. Individual computational agents have bounded rationality, bounded resources for problem solving and bounded both domain and didactic expertise. The primary contribution of the DEE is that it makes it possible to combine resources of different intelligent nodes. The combined abilities of the agents transcend their individual abilities so that the scope and the quality of the teaching activity they perform as a group may increase considerably.

Another significant problem that often arises in computer assisted learning is the incompatibility between the computer knowledge representation and the students' mental models. Discrepancies in conceptual models often make the students unsusceptible to following the computer guidance. The DEE permits different knowledge representation schemes and knowledge perspectives. This makes it possible for the system to capture non-trivial solution paths and exhibit more extensive diagnostic policies. The architecture of the DEE fits in the open-systems model [1]. Open systems have no fixed boundaries and are easily extendible with new knowledge bases or intelligent agents.

The main intention behind the DEE is to support the social activity in the learning process. Interaction between the students participating in the network may simulate their motivation. Since cooperation is an integral part of social interaction [2], a group of students may achieve more by working together than by working alone. The grouping of students may appear on the basis of: common interests, common goals, unsolved problems, similar behaviour and so on.

One of the major requirements to the educational systems is their adaptation ability. In the computer assisted learning systems the only way to obtain the adaptation abilities is by using and developing the user model. In the DEE, adaptation abilities reside in the multiagent organization which may be changed dynamically in the course of teaching. In this case the user model is built by several agents, playing different roles and functions.

Repeatedly changing needs of the students may be met in the DEE by dynamically rearranging the problem solving structure, the structure of roles and priorities. Furthermore, the multiagent structure is tolerant to the failures of single agents and degrades its performance gracefully.

3. Distributed Educational Environment

The DEE is organized as a network of autonomous intelligent nodes. The distribution can arise because of spatial distance, deductive difference or semantical distance. Each student may participate in the network by means of particular nodes supplemented with teaching abilities and didactic expertise. These nodes are called 'tutors'. Out of the network context, the tutor node may be considered as a single intelligent tutoring system [3]. Besides the tutor nodes the network may include other types of nodes: problem experts, teaching experts and so on.

The interface between the network and the student is realized through the tutor node. Because of bounded resources and bounded rationality this node is not able to deal effectively with all the problems to be solved. It is more reasonable to use the knowledge and problem solving skills of multiple nodes, each of which handles some part of the total problem. In the case of the functional and knowledge specialization among the nodes this approach guarantees for the better use of the network resources. In the DEE the initial task allocation is done by the tutor node. After that each node manages its activity on its own, allocating subproblems of the original problem to other nodes.

Each node in the network has an extended blackboard-based architecture [4,5]. The node blackboard is intended to contain only dynamically local context of computations. The global context of the current situation is maintained by global blackboards associated with each student. The student blackboards play twofold function. On the one hand, they contain all temporary hypotheses, current goals and active data received by different nodes. In this way the multiagent planning and synchronization is supported. On the other hand, student blackboards play the role of the students models and action spaces because of their purpose to contain all the necessary information about the students.

The student blackboard is divided into two areas. The first one, called student area, represents student's belief model. Belief changes occur as implications of some student or node action. The second area, called network area, contains network goals, partial solutions, current hypotheses, plans and so on. All the nodes are able to obtain and modify information from the network area.

Belief revision in the student area is managed by the tutor node associated with the student. The student may exchange messages with other students communicating with their tutor nodes. After such communication the tutor nodes are responsible for the subsequent belief revision in the student areas. Another case of belief revision arises when some actions are taken by the network. One category of actions may modify problem space, other actions are connected with the student, a third category of actions change network organization.

Besides the monitoring the student area the tutor node is also responsible for the monitoring of the network area in the student blackboard. All the read and write data-access requests from the other nodes are handled by the tutor node. Since the different nodes are not guaranteed to modify the blackboard uninterruptedly, the tutor node has to ensure the syntactic and the semantic integrity of the blackboard information. This integrity is achieved by supplementary labels for each blackboard element denoting the nodes the element is associated with. As an example, the plans in the network area are labelled with the addresses of the nodes executing them, the hypotheses are labelled with the addresses of the nodes proposed to them and so on. If some modification in the blackboard is made all the nodes it involves are informed by the tutor node. In such way the redundant work is avoided and the node processing power is concentrated in more perspective solution paths. Another advantage of such labelling mechanism is that it provides the network with the global view of the current situation. Given such global view, many conflicts between the nodes may be avoided and better distribution of the processing load may be achieved.

The plan synchronization is carried out by the tutor node. Since the plans of different nodes are connected by temporal and spatial resources and causal relations, interaction in the plans must be controlled. When incompatibility between the plans occurs the tutor node may postpone some plan execution or reallocate the task for replanning to another node. This is possible because of the tutor node ability to manage the student blackboard content. After the modification of the blackboard plans the nodes executing these plans must rearrange their activity according to the modifications. In such a way the tutor node may not only keep a watch over the activity in the network but control this activity. The central role the tutor node plays is relative to its student blackboard. They may be other tutor nodes and other students in the network. In this case all the tutor nodes have the same authority and conflicts between them must be resolved by negotiations [6].

4. Communication protocol and internode interaction

A convenient method for internode communication is contract net protocol [7,8]. It provides powerful mechanism for task allocation, since it permits a more informed choice from among the alternative nodes to which tasks may be allocated. The key element in the task allocation is the negotiation procedure. The negotiations involve two nodes which are called manager and contractor. The manager generates a task to be done and the contractor has agreed to perform the task. The manager announces the existence of the task to all potential contractors. Each contractor evaluates its own level of interest with respect to the type of the task and the available resources. If the task is found of sufficient interest the contractor informs the manager about its readiness to perform the task. The manager may choose from among the several potential contractors. It selects those contractors which better match the task requirements. A contract is thus an explicit agreement between a node that generates the task and a node that executes the task. The nodes are not constrained to be only managers or only contractors. It is possible for a node to be simultaneously both manager and contractor for different contracts. As an example, in the process of executing the task the contractor may deliver parts of the task to another contractors. In this case this node is a contractor in respect to the whole task and a manager in respect to the parts of the task.

In the DEE the initial tasks are generated by the tutor node. The announcement of these tasks is carried out by placing them into the student blackboard. After negotiations the tasks are allocated to contractors and the contractors' names are placed into the blackboard. In this way all the nodes are acquainted with the network activity. Furthermore, by virtue of this publicity, several contractors in a given contract may share partial results and communicate between each other. In the conventional contract net protocol such communication is not possible because the task allocation information is inaccessible.

The mechanism of task allocation by global blackboard helps the work duplications to be avoided. As an example, a node may check the student blackboard whenever it receives a new task. If such a task is performed by another node, the first node may simply copy the results.

The global blackboard approach is also well suited for the communication management. In the complex networks with limited communication bandwith, when each node must communicate with many other nodes, the communication policy is of great importance. By examining the student blackboard the nodes may reason about each others' present activities. It allows the nodes to better predict the effects a message will have on the

other nodes. Predictions lead to reducing communications because only necessary data need to be communicated. In fact, the student blackboard plays the role of the activity map in which all the points of task and result sharing are marked. The nodes use the activities allocation information to recognize and establish communication links, to discover how activities may be reordered to avoid harmful interactions and to promote helpful interactions.

More interesting communication problems arise when there are many students and many tutor nodes in the network. While the tutor nodes have higher priority in respect to other kinds of nodes, the priorities of all tutor nodes are equal. This requires more complex decisions about the interaction between the students and the tutor nodes. The DEE allows the students to communicate between each other, to form groups of interests, to pursue a goal by common efforts and so on. In such a way different student's viewpoints may be analyzed and a common cognitive space may be constructed. The ability of the DEE to coordinate different users resembles to a great extent the participant systems [9]. In contrast to conventional participant systems, the DEE has its own problem solving abilities and domain expertise. This makes it difficult to coordinate the activity of the network and the students groups. In the DEE three heuristics are proposed for conflict resolution between the network and the students:

1. Group formation. When conflict arises between the students the current individual student goals may be changed to equivalent ones which promotes cooperation.
2. Group destruction. When conflict arises between a student group and the network the students are given competitive goals.
3. Changing the focus of attention. When internal conflict arises between the tutor nodes the students are given other problem situations. In such a way plans and preferences of the tutor nodes are changed.

5. Conclusions

In this paper we have presented a new approach for computer assisted learning. This approach incorporates features of the intelligent tutoring systems and distributed problem solvers. It has been devised to permit the multiperson - multimachine interaction and to stimulate the social activity of the students. Several perspectives for further research remain. A mechanism for internode cooperation must be developed. On its basis the tutor nodes may control the student groups and influence the behaviour of each student. For these purposes the economic and game-theoretic models may be useful [10,11].

References

1. Hewitt, C.: Offices are Open Systems. In: B. Huberman (ed.) Ecology of Computation, 5-24. Amsterdam: Elsevier Science Publishers, North Holland 1988
2. Axelrod, R.:The Evolution of Cooperation. Basic Books 1984
3. Sleeman, D., Brown, J. S.: Intelligent Tutoring Systems. New York: Academic Press 1982
4. Hayes-Roth, B.: A Blackboard Architecture for Control. Artificial Intelligence 26, 251-321 1985
5. Durfee, E. H.: Coordination of Distributed Problem Solvers. Boston: Kluwer Academic Publishers 1988
6. Rosenschein, J. S., Zlotkin, G.: Negotiation and Task Sharing Among Autonomous Agents in Cooperative Domains. In Proceedings of the 1989 International Joint Conference on Artificial Intelligence, 912-917. Morgan Kaufmann Publishers 1989
7. Smith, R. G.: The Contract Net Protocol: High - Level Communication and Control in a Distributed Problem Solver. IEEE Transactions on Copmuters C-29, 12, 1104-1113 1980
8. Davis, R., Smith, R. G.: Negotiation as a Metaphor for Distributed Problem Solving. Artificial Intelligence 20 (1), 63-109 1983
9. Chang, E.: Participant Systems for Cooperative Work. In: M. Huhns (ed.) Distributed Artificial Intelligence, 311-339. London: Pitman 1987
10. Rosenschein, J. S., Genesereth, M.R.: Deals Among Rational Agents. In Proceedings of the 1985 International Joint Conference on Artificial Intelligence, 91-99 1985
11. Malone, T. W.: Modeling Coordination in Organizations and Markets, Management Science 33 (10), 1317-1332 1980

A change of view

The parameter projection problem in threedimensional view setting

Eric Brok[1] and Peter van Splunder[2]

[1] Open university, Faculty of Engineering, Box 2960, 6401 DL Heerlen, the Netherlands.
[2] PTT Research, Insitute for Applied Social Science Research, Box 421, 2260 AK Leidschendam, the Netherlands.

Abstract. An experimental on screen control panel for 3D view setting is described. The design requires only a regular 2D mouse input device. The user sets the desired viewpoint by direct manipulation of metaphoric icons, which is more facile and natural than by setting several slide control bars.

This specific control panel is also adressed as an illustration of a general interface design issue: the internal technical parameters of the computer program must be projected onto the user interface with great care to avoid confusion in the user.

Keywords. User/machine systems, human factors, interface design, computer graphics, 3D graphics, interactive graphics, interaction techniques, input devices, virtual controllers, rotation control, view setting, parameter projection problem.

1 Introduction

1.1 The parameter projection problem

Usually he functioning of a computer program is determined by a set of internal parameters. Many of these parameters are to be controlled by the end user. Seen from the view of a computer programmer, it would be most obvious to expose these parameters directly to the user. But in many cases the technical parameters by itself are not organized in a way that the user can grasp easily. The user probably relates to the situation at hand by very different conceptual entities. In short there is a gap between technically significant parameters and psychologically significant parameters. To facilitate human-machine communication these two sets have to be projected carefully onto each other. We will call this issue the *parameter projection problem* (PPP). We will discuss this issue further by an example interface.

1.2 Threedimensional communication

In general, computer systems communicate information from and to people. This information of course can take many forms, one of which is graphical in three dimensions. It can be graphical by nature (e.g. a landscape) or by modelling (e.g. the results of a factor analysis). Communication of threedimensional information is important in various fields of interest such as artistic illustration, technical design, architecture, city planning, math, chemical engineering, medical diagnosis and so on. These fields have developed numerous practices of education and training. So it is clear that 3D graphics are important to educational settings as well. Of course the students and trainees should not be forced to divert their attention from learning the prime subject to figuring out the user interface. Therefore particularly in education the PPP issue deserves explicit attention.

Display of 3D information usually involves 3D input from the user. Often the content matter itself is generated by the user such as in drawing and design. But even when the computer system generates the scene, the user might have to communicate 3D information such as the desired viewpoint from which to look at the scene. This paper concerns this type of spatial input which we will call *view setting*.

1.3 Optical projection issues

Before turning to the PPP-issue, we want to make clear that in 3D graphics there is also an optical projection problem. First threedimensional objects have to be projected onto the flat screen. Mathematics (stereometry) deals with that quite well. Second and more problematic is the twodimensional on-screen manipulation of threedimensional objects. Simply pointing at an object can be already ambiguous (Bier, 1990). It is now widely acknowledged that creating and manipulating 3D objects is a complicated task (e.g. Veniola, 1993). In recent years, as 3D graphical systems became increasingly available and cognitive ergonomics gained attention, researchers turned to the issue.

1.4 Related studies

One way to deal with this issue is to develope advanced mechanical interaction devices. Datagloves, 3D glasses, 3D mouse devices (e.g. Venolia, 1993) and the like seem to bring truly threedimensional interaction literally within reach. Existing CAD/CAM applications often use hardware dials to rotate threedimensional objects. This is another way of diverting interaction away from the flat screen. We acknowledge those approaches, but this paper will discuss an interface that only requires on-screen interaction with 2D mouse. (The design of this interface however could easily be adapted to true 3D input.)

Chen, Mountford and Chellen (1988) describe several virtual controls for rotating 3D objects. Most promising seems what they call the 'virtual sphere' (which they also compare with a resembling control originally described by Evans, Tanner & Wein (1987)). This control is a virtual track ball that can be manipulated by mouse cursor. Rolling it up or down makes the selected object pitch (see also table 1). Dragging sideways makes the object yaw. And (unlike physical trackballs) dragging along the circumference makes the object roll. However promising, this control concerns only rotation of a single object. Object selection and translation throughout the scene might call for another approach. The current paper therefore explicitely focusses on the *combined* task of translation and rotation.

2 The example program

2.1 Turning tables in PLEXI

The control panels we will discuss are part of a prototype application program. We will first take a quick glance of this program called PLEXI[1] itself. PLEXI enables the user to furnish a threedimensional room (see figure 1). Because arranging furniture mainly concerns movements along the floor, this interaction takes place in a 2D ground plan window. To add furniture to the room, the user can pick pieces of furniture from a pallet (the 'storage room') by mouse and drop them somewhere on the ground plan. Once present in the room, the pieces can be moved and turned around by dragging. They also can be deleted, copied and assigned color.

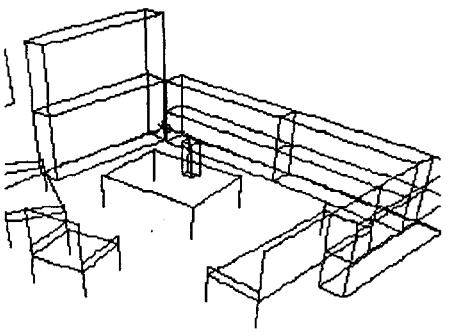


Fig. 1. A room with a view.

The program is unlike a mere drawing program because it checks all user rearrangements for violations of certain syntactical constraints. For example a chair can be moved under a table, however excluding its back and not at a corner of the table.

[1] If you wonder why the program is called PLEXI this might interest you. In a realistic 3D picture, objects obscure each other. Simulating this effect in computer graphics requires considerable computing effort, in particular when objects can obscure their parts mutually. We did not want to focus on the mathematical intricacies of this issue which is called *hidden surface removal*. Instead we simply designed the program to picture wireframe furniture only. And in a corny mood we decidedly claimed that the program was still perfectly realistic; it just is custom made for those fancy interior decoraters who furnish entire rooms by *plexiglass* furniture only.

The room is also continually depicted in a seperate '3D window'. (Of course the window itself is 2D, but a 3D structure is projected onto it.) So through this window the user can view the room from various angles and distances. To obtain a certain view on the room, the user must communicate the desired viewpoint to the computer. For this, PLEXI supplies two very different control panels, on which this paper will focus.

2.2 The parameters

To appreciate the PPP-issue in this example some technical details should be noted. The program is written in Pascal. The programming environment came with several special purpose library modules. One of those modules calculates stereometrical projections of 3D wire frame structures onto the screen. It forms an opaque module that is not accessible by the programmer. In order to generate a picture the module needs two kinds of input information:

- A complete description of the scene by coordinates in a virtual 3D space;
- Several parameter values to determine the view on the scene.

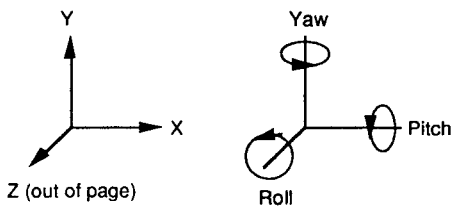


Fig. 2. The axes and rotations as defined by the library module.

These parameters are of key importance to the current discussion (see table 1 and figure 2).

Table 1. The parameters that determine a view.

Parameter	Effect	Reference
Yaw	Rotate camera around...	y axis
Pitch	Rotate camera around...	x axis
Roll	Rotate camera around...	z axis
Scale	Closeness of camera to...	center of space
MoveX	Displace rotation center along...	x axis
MoveY	Displace rotation center along...	y axis
MoveZ	Displace rotation center along...	z axis

3 Two different views

3.1 The slide bars control panel

Seen from the viewpoint of a programmer it would be most obvious to expose these technical parameters directly to the end user. This approach results in the control panel in figure 3. There is a seperate slider for each parameter. The yaw, pitch and roll vary from -180 to +180 degrees. The move parameters vary from -100 length units to +100 length units.

Using this control panel we experienced some problems. First of all, the names of the parameters poorly explain their effects, especially for non English native speakers. And any better names or translations are hard to imagine. But most confusing of all, the different parameters interact mutually. E.g. the effect of rotation around y axis depends on the displacement of the center of the space. This is of course stereometrically coherent but can be psychologically hard to interprete and predict. The control panel does not support a consistent mental model about the relative effects of each of the parameters when combined. Related studies too suggest that users often find cummulative rotation around all three axes hard to monitor and control (Chen, Mountford and Sellen, 1988). And problems become only worse when the room content is so complex that each redrawing takes considerable time (e.g. two seconds). Users will resort to larger parameter changes at a time. As a result the rotations and displacements in the picture will be too jumpy to communicate the effect of the changes directly. Moreover the user has to *compose* a certain viewpoint by setting the seperate parameters one at a time. Intermediate pictures are often confusing. In short the user gets easily lost in trial and error behavior without proper feedback.

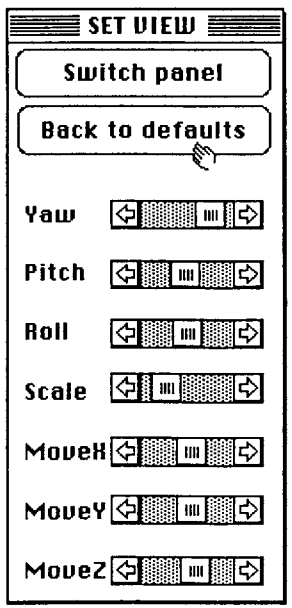


Fig. 3. The slide bars panel.

3.2 The icon based control panel

We decided to change our view from the programmers' to the users'. For the moment we forgot all we knew about the 3D Pascal module. Another control panel was designed from scratch that would make more sense to the user. The new design was based on two general interface design concepts: *methapors* and *direct manipulation*. Metaphoric interfaces explicitly use everyday objects and events from outside the computer realm to support computer based tasks. A classic

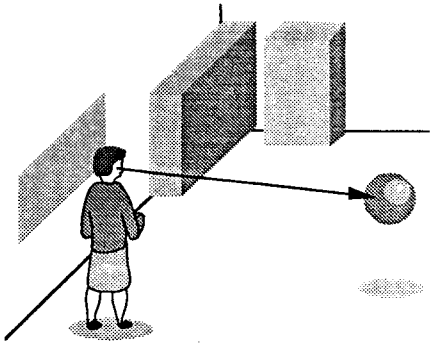


Fig. 4. Throwing a look.

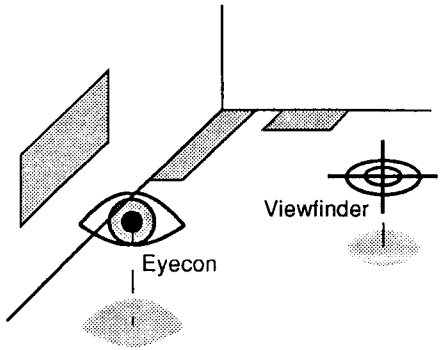


Fig. 5. The first draft.

example is the desktop metaphor of the Xerox Star system (Smith, Irby, Kimball and Harslem, 1982) and the Apple Macintosh. Metaphor guides the construction of a mental model about the functioning of the computer program. Metaphor is a powerful resource for design because it exploits prior knowledge; by analogical reasoning even novice users will know what to expect from the system and how to respond to specific situations (Gardiner and Christie, 1987). A second and strongly related design concept is direct manipulation. It requires visual representation of the domain of discourse (e.g. by metaphoric icons). Then the user can perform operations by pointing, dragging and editing objects. The benefits of direct manipulation as opposed to command lines or menus, include rapid learning, less errors, high user satisfaction and encouragement of explorative user behavior (Schneidermann, 1991).
The new PLEXI control panel involves direct manipulation of methaporic icons. It introduces a third-party perspective in which the user sees herself, watching a part of the scene (see figure 4). The user can then simply displace the icon representing herself to the desired viewpoint and direct her look towards a certain spot.

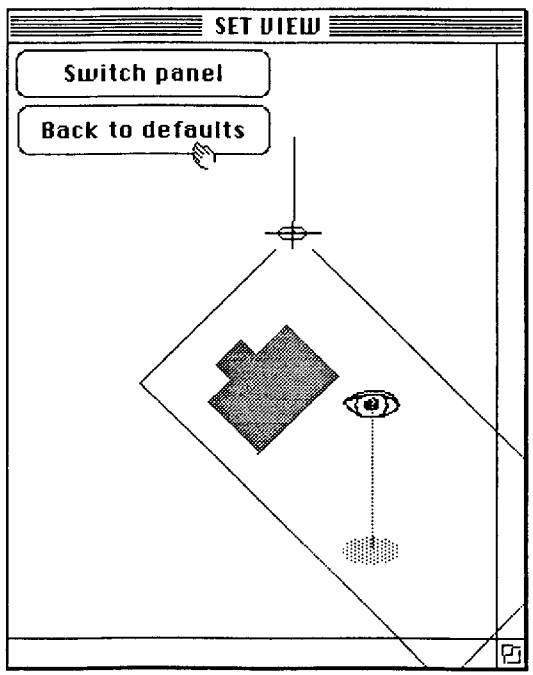


Fig. 6. The implemented 'eyecon' control panel.

Actually this is not quite that simple because both viewer and focus spot should be manipulated in all three dimensions. Figure 5 shows our first draft. Viewer and spot are represented by methaporic icons, which we will call Eyecon and Viewfinder respectively. The drawing of a person is replaced by a more abstract Eyecon for three reasons. First, the picture of the viewer should obscure at least of the scene as possible. Second, a real person would not be able to float over the ceiling, underneath the floor and behind the walls of the room. Third, unlike a person or a camera, Eyecon is a spheric object. Therefore it is less disturbing that the icon itself does not threedimensionally turn to the viewing direction.

For technical reasons we had to simplify this draft to the implemented version, which is depicted in figure 6. This control panel looks like the actual room. On the floor are the continually updated contours of the furniture arrangement. These can not be manipulated in this panel. The position of Eyecon relates to the 'camera' viewpoint in space. The relative position of Viewfinder to Eyecon specifies the direction in which the 'camera' looks. So the view is always from Eyecon to Viewfinder. The user can modify the horizontal position of an icon by dragging its shadow along the floor in any direction. The icon itself moves automatically along, keeping straight above its shadow. The icon can be dragged only vertically which determines the height. (Note that this restriction to vertical movement is necessary. Otherwise users might drag the icon in an oblique direction and it would be impossible to decide which combination of x, y and z-movement they intended.) In this way the user can specify a viewpoint by one to four meaningful movements instead of one to seven technical parameter changes.

Once the eyeconic control panel was devised, its parameters had to be projected somehow to the seven technical parameters which the Pascal module required. This is the actual PPP issue (see figure 7). The position of the shadow of Viewfinder along the floor determines the *x-z displacements* and the distance from Viewfinder

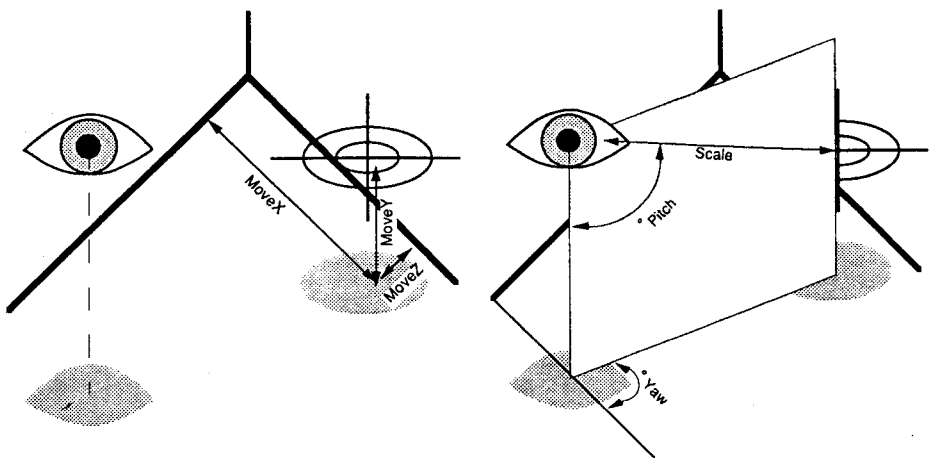
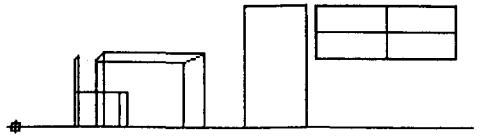
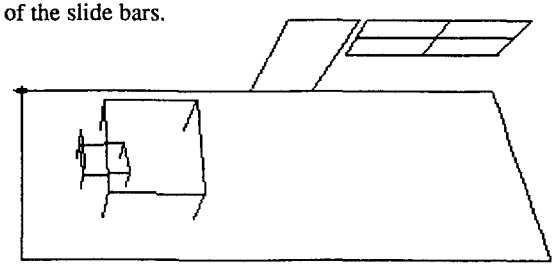


Fig. 7. Actual projection of internal parameters onto interface semantics.



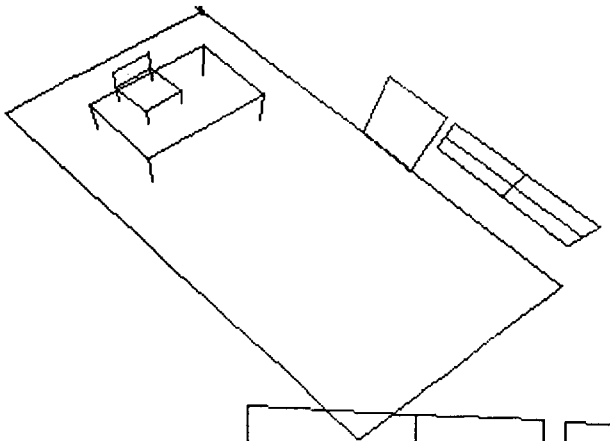
1

Table 2. (Two-page spread.) Four views and their control settings. From left to right: four views, the respective arrangements of the eyecon panel and the respective settings of the slide bars.

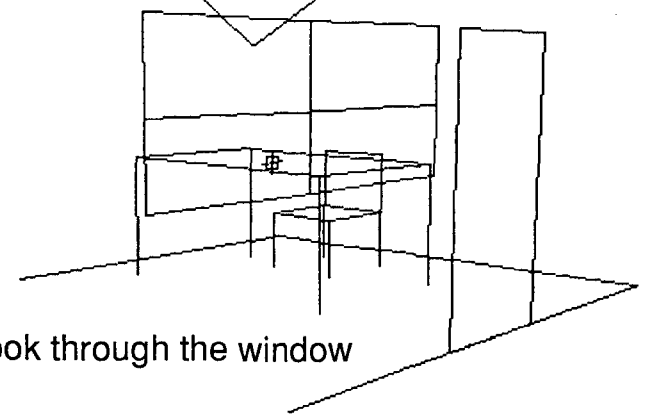


2

Basic manipulations

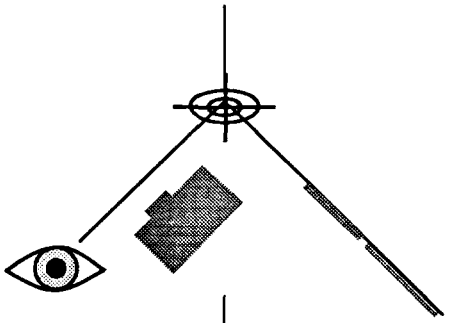


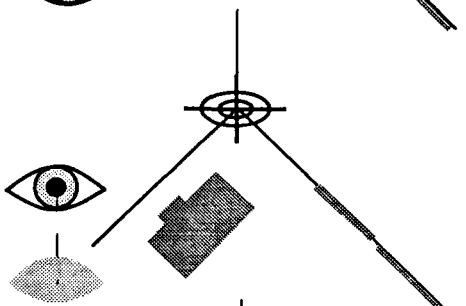
3

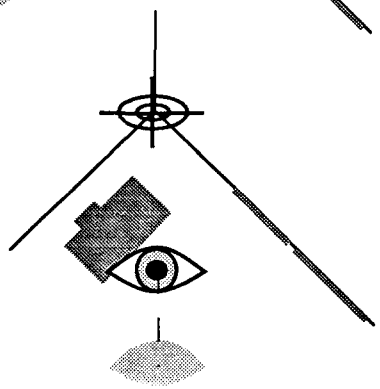


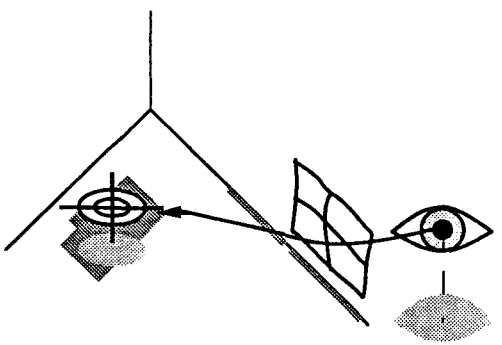
4

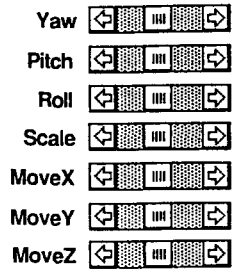
A look through the window

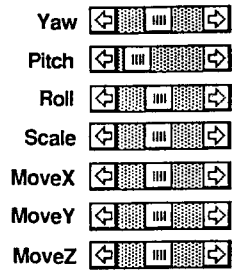


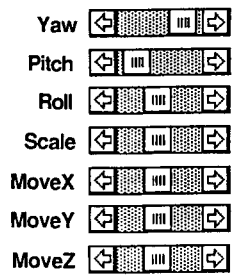


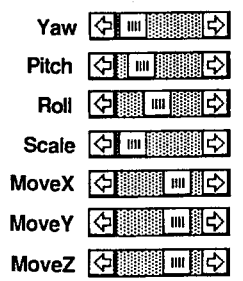












to its shadow determines the *y-displacement*. Thus Viewfinder represents the center of space. The direction in degrees from Eyecon to Viewfinder along the floor determines the amount of *yaw*. The difference in heigth (icon-to-shadow distance) between Eyecon and Viewfinder determines the *pitch*. The *roll* parameter is not implemented but could be assigned to local rotation of Eyecon (by dragging at one of its sharp ends, see figure 8). *Scale* was calculated as the 3D-Euclidian distance between Eyecon and Viewfinder, multiplied by a heuristic factor.

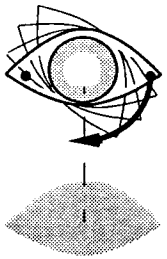


Fig. 8. Possible mouse input of roll parameter.

Of course this parameter projection is both psychologically and mathematically rather unsophisticated, but merely serves as an illustration of the general problem of parameter projection ('3P in 3D').

Table 2 shows in a glance how the visual appearance of the two control panels relate to the changing 3D view. The first three views show merely how seperate parameters effect the view. View no. 4 however serves to demonstrate the clear advantage of the eyecon panel in combined translation and rotation. Would you be able to guess the slide bar settings right to get this peep through the window?

4 Experiment

In order to learn more about the usability and learnability of the eyecon panel, a small scale experiment was carried out. This experiment was designed to compare the eyecon panel directly to the slide bars panel regarding learnability by novices.

4.1 Method

Subjects. Five male and five female right handed adult subjects were tested. Five of the subjects had an academic degree, three of which in mathematics. All subjects were familiar with using a mouse device. They had no prior experience with PLEXI or any other 3D graphics systems.

Apparatus. The experiment was run on an Apple Macintosh LCII computer with color display and a one button mouse. The software (PLEXI itself and the experiment control) was written in Pascal. Accuracy was recorded on line. Task completion time was recorded by a hand held stopwatch. The roll parameter was hidden in the slide bar panel because it was not implemented in the eyecon panel either.

Tasks. Subjects were asked to perform a series of matching tasks. On the screen they were presented a plain room seen from a certain position. Then they were shown a picture on paper of a certain desired view and they were asked to match the screen picture in angle and size to the paper copy. Both target and screen picture were wireframe rendered. Subjects decided themselves when the match was sufficient.

Each series consisted of four matching tasks:

task 1	from view 1 to view 2 (see table 2)
task 2	from view 1 to view 3
task 3	from view 1 to view 4
task 4	from view 4 to view 1

Both task completion time and accuracy of match were measured. Accuracy was calculated as the sum of the squared differences between the two views on the parameters yaw, pitch, scale, moveX, moveY and moveZ. Subjects were not allowed to switch to default view settings during the tasks.

Design. According to a within subject design, each subject performed the same task series with both control panels. Order of control panel was counterbalanced: half of the subjects used the icon based panel first, the others used the slide bars panel first. This condition was equally divided over sex because gender might relate to ease or style of spatial reasoning. Verifying such effects however would require a much broader study.

Procedure. All instructions were provided by the test leader. In addition to a general introduction to the procedure, screen layout and upcoming tasks, each session consisted of *twice* the following sections, once for each control panel:

- Introduction to the control panel
- Self-discovery with this panel for five minutes
- Supplementary explanation by experimenter if neccessary
- Task 1 to 4 using this panel

To make sure that subjects really tried to understand each control panel during the discovery period, they were asked to explain verbally how each icon or slide bar related to the 3D picture. If this revealed serious a priori misconceptions (which was rare) some supplementary explanation was provided by the test leader. Each session took about half an hour and tasks were hardly repetitive so there was no need for a break. At the end of the session the subject was asked a 1 to 10 rating for the usability of each control panel.

4.2 Quantitative results

Performance in style and completion time differed even more between subjects than we already expected. Figure 9 to 11 show the quantitative results. Figure 9 shows that averaged over subjects all tasks were completed fastest with the eyecon panel. The difference is particular large regarding task 3. Note the relatively large standard deviation. Figure 10 shows accuracy of task result. For the simpler tasks (1 and 2) the slide bars gave least error. For the more complex tasks the eyecon panel gave least error. Figure 11 shows that the average rating was higher for the eyecon panel than for the slide bars panel.

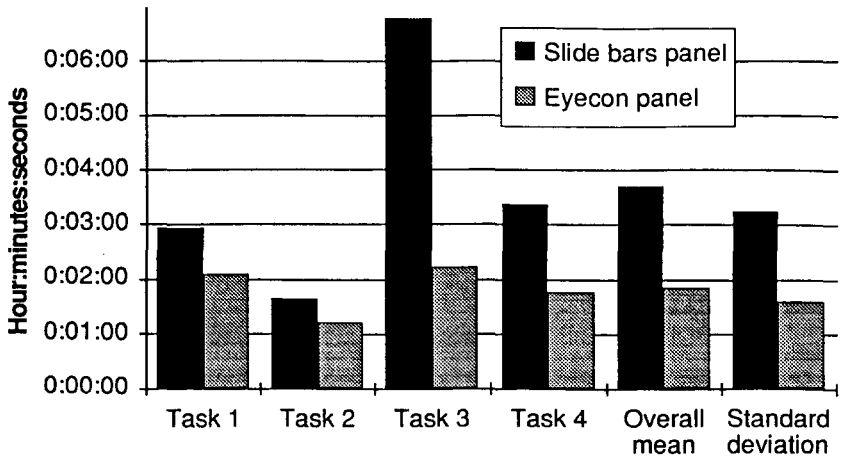


Fig. 9. Task completion time averaged over subjects.

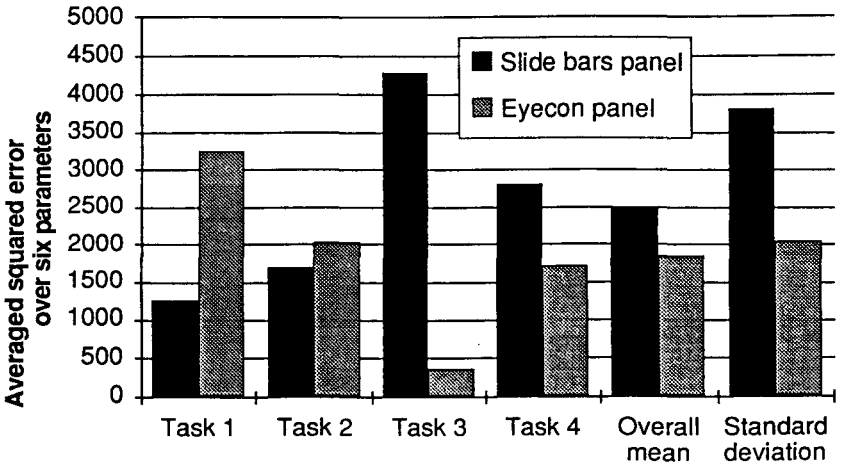


Fig. 10. Error in result view averaged over subjects.

4.3 Discussion

Different strategies. Completion time very much depended on choice of strategy. The 3D window always depicts the center of space (which is center of focus and rotation as well). But because this center point is projected onto the 2D screen, it does not reveal its actual 3D position compared to the room. Often the center point only appears to coincide with the target part of the room, and drifts away when rotating the room around. Therefore, using the slide bars panel, a strict strategy should be used to first set the center point in all three dimensions which requires watching the room from at least two orthogonal views. And not until then the room should be rotated to achieve the desired angle of view. Only one of the subjects developed this strategy. When using the eyecon panel there is no need for such a cumbersome strategy because the 3D position of the center point is explicitely represented by the Viewfinder.

Also the two control panels proved to call for very different strategies according to the use of feedback information. Using the slide bars panel it proved very hard to predict the effect of each slide shift; the subjects continually watched the 3D window to recieve indispensable feedback. But using the eyecon panel users should focus entirely on the control panel itself. While moving the icons around it proved unneccesary to interprete the intermediate results. It can be even confusing; when no part of the room is between Eyecon and Viewfinder the 3D window is (of course) left almost blank. Thus the control panel itself explains the situation better than the 3D output window. From this observations we can conclude that the eyecon panel has a clear advantage to the slide bars panel when the room content is so complex that each drawing takes considerable time.

Task difficulty and completion time interaction. We expected an interaction effect between control panel type and task difficulty. Tasks 1 and 2 did not require moving the focus spot at all. So in the eyecon panel the Viewfinder should not be displaced and in the slide bars panel the moveX, Y and Z slides should not be scrolled. Therefore both panels were expected to be quick and easy for these tasks. The advantage of the eyecon panel was expected to emerge only in tasks 3 and 4. Results for accuracy (figure 10) actually revealed such interaction. The results for task completion time did not, because task 1 and 2 did not turn out to be as simple as was expected, which depended strongly on first action choice. Although tasks 1 and 2 did not *require* changing the focus spot, subjects still often did. Subjects that scrolled the move slide bars at the beginning of task 1 or 2, were immediately lost in space. Likewise though much less severe, subjects diverted from the right track when they first started moving the Viewfinder first instead of the Eyecon. So a rather accidental first choice resulted often in a large task completion time. As an afterthought we regret that the 'back to defaults' button was disabled during the experiment.

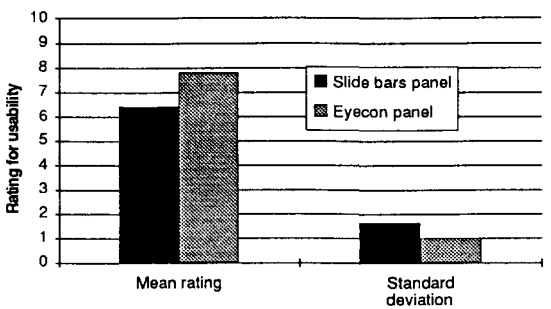


Fig. 11. Subjective rating of ease of use averaged over subjects.

We are primarily interested in the usability of the eyecon panel, so user difficulties with the bars panel will not be discussed in detail any further. The major drawbacks for the eyeconic panel proved to relate to three seperate issues which we will now adress briefly.

Dragging technique. In the experiment version of the eyecon panel, an icon can only be dragged along the floor by dragging its shadow. The unnaturalness of this technique proved persistent; some subjects repeatedly tried to drag an icon itself horizontally even when they already rationally knew they should drag its shadow. Moreover, if an icon lies exactly on the floor, it obscures its shadow. To move the icon along the floor the icon should be lifted first to reveal its shadow. Subjects found this very annoying. Fortunately these techniques are not intrinsic to the eyecon panel. In response to these observations an alternative version of the panel was implemented. In this version a modifier key is added to the scheme. The icon can be dragged along the floor immediately. It does not matter whether it is dragged itself or by its shadow. To move the icon further or closer above its shadow the modifier key must be depressed during the dragging. This scheme, which is only a first and yet to be tested alternative, brings along extra movements (it involves a second hand) but probably will prove more natural.

Relating different views. The second issue is inherent to the used metaphor of a person walking around an object (as opposed to e.g. turning an object around in your hands). Remind that the control panel shows a static third party perspective in which both viewer and object are represented. So if the room is watched from behind, left in the control panel becomes right in the 3D window. (This also happens when using the slide bars panel.) This extreme example demonstrates that the user must be able to relate the directions in the panel to the directions in the 3D window. Fortunately most subjects succeeded in this quite well.

Cluttering windows. PLEXI supplies a 2D ground plan window and a 3D view window. The eyecon control panel inroduces yet another view-like window. Too much windows can be cumbersome. Therefore we think it would be wise to integrate some of these windows. The control panel intrinsically can not coincide with the 3D view because a viewer can not see herself standing in her own view on the scene. But the control panel could coincide with the ground plan, simply by allowing the user to create and rearrange the contours of furniture directly on the floor in the control panel.

5 Conclusions

Regarding to a small scale test and opposed to the slide bars panel, the eyeconic control panel clearly shows considerable improvement in learnability and ease of use without loss in accuracy. Subjects generally rated the eyecon panel higher in ease of use. Observations gave rise to several ideas for further improvement of the eyecon panel such as change of dragging technique and integration with the ground plan window.

Whenever one writes a computer program based on existing library modules it can be very tempting to duplicate the internal organization of parameters directly to the user interface. In general and certainly in our example this can lead to communicational problems. Technical parameters can differ from psychological entities and they can interact in difficult to predict ways. Then a deliberate change of view can be necessary to design a more user oriented organization of parameters.

References

Bier, E. (1990) Snap-dragging in three dimensions. In: Proceedings Workshop on Interactive 3D Graphics. ACM/SIGGRAPH, 183-196.

Chen, M., Mountford, S. J., & Sellen, A. (1988). A study in interactive 3D rotation using 2D control devices. Computer graphics 22, 4, 121-129.

Evans, K. B., Tanner, P.P. & Wein, M. (1981). Tablet based valuators that provide one, two or three degrees of freedom. Computer Graphics, 15, 3, 91-97.

Gardiner, M.M. & Christie, B. (eds.). (1987). Applying cognitive psychology to user-interface design. Chisester: Wiley.

Schneidermann, B. (1991). A taxonomy and rule base for the selection of interaction styles. In: B. Schakel, S. Richardson (eds.) Human factors for informatics Usability, 325-342. Cambridge: University Press.

Smith, D.C., Irby, C., Kimball, R. & Harslem, E. (1982). The Star user interface: an overview. Submitted to the AFIPS 1982 National Computer Conference.

Veniola, D. (1993). Facile 3D direct manipulation. In: Proceedings Interchi '93, 31- 36.

Partners Dialogue as a Method for Non-Formal Problem Solving

Igor Chmyr

Institute of Low Temperature Engineering and Energetics
1/3 Petra Velikogo st., Odessa 270100, Ukraine

Abstract. Man-machine dialogue can be used not only for interface organization but also as a "solver" for those problems that consist of a set of heuristics united by "expert logic". Using a special kind of relationship "stimulus-reaction" we can build a scenario that imitates the expert work. The dialogue problem solver contains a scenario that is an access method to the set of stimuli as described.

1 Dialogue imitation of expert work and "dialogue solver" organization

Practically all the methods used in knowledge delivery or educational systems are non-formal, being invented and realized by experts.

We can imitate the expert work by computer if we are able to present it in a dialogue form which will satisfy the following requirements:

- the computer remains an active partner during all the dialogue process and at every step it shows the passive partner a stimulus according to the scenario;
- every stimulus contains a direct or indirect form reaction that is expected;
- for every applied problem in scenario building there are a finite number of stimuli and reactions.

Therefore, during the dialogue imitation of the expert work, the computer had to engender an aim-directed sequence of stimuli in accordance the scenario.

We will understand by scenario the method that allowed us to generate the ordinary stimulus after the current reaction was recognized.

One of the key ideas is, that a scenario does not "calculate" the ordinary stimulus, but finds the stimulus in the certain direct access memory – *StimulStore*.

The scenario has to support two functions:

- to store an "expert logic" of problem solving;
- to generate a current stimulus index for *StimulStore*.

And in the previous passage the scenario is a certain access method, which will be marked as *DiAM*. A various number of applied scenario analyses allows us to make a conclusion that in general it must have a network organization. Every node of this network is corresponding to the dialogue process step, and every step — to the elementary dialogue contact act:
- generation of the stimulus and transmittion to the passive partner;
- reaction perception;
- reaction recognition and current stimulus index definition.

DiAM network may be described by various mathematical methods (Petri nets, for example), but it is more constructive to describe *DiAM* as an abstract relational database. This description has two important advantages:
- it does not prevent a complicated node elements classification to engage;
- it converts the *DiAM* into data which is simple to edit.

The dialogue problem solver is biult on *DiAM* base and contains the following components:
- *stimuli permanent memory* (a direct access memory that stores an indexed set of stimuli descriptions);
- *dialogue process temporary memory* (this memory stores a sequence of passive partner reactions, which is used by certain type of *DiAM* nodes);
- *dialogue access method* (database that stores a problem solving method. For every input reaction *DiAM* provides an access to a single indexed element of *StimulStore*);
- *dialogue processor* (a resident process that provides step by step scenario interpretation);
- *demons-procedure library* (set of external procedures that realizes a function cannot be covered by *DiAM*)

2 Stimulus structure

The active partner generated stimulus and the answering reaction are in close relationship. Every stimulus predetermines several reactions that take place in that stimulus. This assumption is valid with the question-answering logic results. In accordance to this logic, every question predetermines the answer.

The stimulus is interpreted via two channels: the video-channel and the audio-channel. It exists as a chain of video and audio-objects.

It is natural to assume that one of the video-objects can figure as a reaction. In this meaning, all video-objects divide into three classes:
- an object that is not a reaction. This object is characterized by it's place in the chain only;
- an object that is a direct reaction. This object can be chosen as a reaction. It is characterized by its place in the chain and individual identificator;

– an object that is a mediate reaction. After being chosen this object must be added by data. It is characterized by it's place in the chain, individual identificator and meaning.

The video-object visual expression is given by two grope of descriptions:

– attributive descriptions;
– non-attributive descriptions.

Non-attributive descriptions are a binary file which is associated with the video-object. Attributive descriptions give the mode of this binary file interpretation (video-object location on the monitor, dynamic effects, animation and so on).

3 Conclusions

A complex approach to the building of a system that is oriented to non-formal problem solving by means of the expert dialogue work imitation.

The probable application sphere of this approach-expert work imitation is: knowledge-transmission, education and design. A system which is described on the level where technical implementation of its elements is inessential, but at the same time this description reflects a modern computer architecture.

The Design of Interacting Agents for Use in Interfaces

David Connah

Institute for Perception Research, P.O. Box 513, 5600 MB Eindhoven,
The Netherlands

1.0. Introduction

The central topic for this summer school is man-machine communication in educational systems. My lecture will not refer very much either to communication or to educational systems but I think I can justify its subject matter because of the inclusion in the title of one other small word. The full title of the summer school is 'The *Basics* of Man-Machine Communication for the Design of Education Systems' (my emphasis) and it is my intention to describe a kind of agent which I think has broad potential in man-machine communication generally and in educational systems in particular.

I will not attempt a rigorous definition of what I mean by an agent but I will rely rather on the common-sense, everyday meaning that people give to the word. Broadly speaking agents 'do' things; they are (apparently) capable of acting autonomously. I would like however to repeat the distinction made by Laurel (Laurel 1991) between two varieties of agent. Agents can be seen as acting on behalf of someone else; for example, an estate agent who sells your house for you or an insurance agent. They can also be seen simply as acting in their own interests and most 'biological' agents (people, animals) are in fact acting in this way most of the time. Much of the literature has dealt with the first kind of agent - Apple Guides come to mind immediately - but the second kind is in fact more general and subsumes the first kind. In what follows I am thinking of this more general kind of agent.

The use of the word agent is not new in Artificial Intelligence (AI). I believe that it goes back to the very beginnings of the discipline. What is more recent is the notion of situated agents. If there is a date at which people in AI began talking about situated agents that date is probably about 1984. Many of the ideas which contributed to this idea are however much older than that and have often been drawn from completely different disciplines such as sociology, anthropology and philosophy. 1984 (if that is the right date) was when people in AI began to realise that some of the problems that were causing such trouble in AI might be eased (or even go away completely) if a different approach was tried. Let me start by talking about models of action.

2.0. Models of Action

Until a few years ago, it was almost universally the case that the actions of agents were described and implemented in terms of a planning model. Roughly speaking this means that agents were thought to have goals. In order to achieve these goals they constructed internal models of (part of) the world and then, by reasoning about these models, they constructed a plan of action. In its simplest form such a plan is a sequential list of actions to be taken by the agent. Once the plan had been constructed it would then be executed by the agent usually resulting in some externally observable behaviour or actions. In this model goals, plans, internal representations of the world and reasoning go hand-in-hand to cause the behaviour of the agent. This is perhaps not the whole story; the agent, on this theory, has in some sense to *intend* to act before action becomes possible. This gives rise to a belief-desire psychological model to explain intention. All of these things to a large extent hang or fall together.

Nowadays, however, this is not the only model. There is another theory called situated action theory (Suchman 1987). In this model an agent acts in a way which is 'appropriate' to its situation. Here 'appropriate' means in a way which evolution or learning has engendered in the agent to ensure its survival and the satisfaction of its interests. The 'situation' is a potentially complex combination of external and internal events and states. In this model there is no suggestion of planning, desires, beliefs etc. but there are other ideas that are implied by the model. The most important of these is that agents have to be closely-coupled to their environment. This is so because it is only through their direct perception of the world that they can be aware of their situation and only by this awareness that they can act appropriately. Thus perception is essential for this model but the model also has implications for such things as representation, emergent behaviour, social behaviour and skill.

To some extent the planning model and the situated action model are extremes appropriate to different circumstances. The planning model was developed when AI was seen to be mainly about formal problem solving (chess playing, theorem proving). Situated action is a more recent theory which is principally concerned with all those aspects of an agent's behaviour which relate to the mundane business of surviving, moving about, performing fairly routine tasks and generally 'living a life' as Chapman puts it (Chapman 1991).

It is unlikely that the simple planning model described here is in fact of practical use in any real situation except, perhaps, such relatively marginal activities as playing chess; the world in general changes too fast and too unpredictably to allow plans of this sort to work. People do, of course, plan but their planning activity is much looser than AI planning and only works at a reasonably large granularity. For example one might plan to go to the airport by car but one cannot plan each move of the steering wheel or each gear change since these are dependent on the as yet unknown behaviour of other road users and road conditions. This example hints at

the way in which a loose kind of planning model and a situated action model are complementary to each other.

Many of the things that we want to do with agents can be done by skilled (as opposed to 'intelligent') agents and situated action theory is very well suited to the description and implementation of such agents. The rest of the lecture is about situated agent theory and about some of the implications that the theory has for the design of situated agents. One thing that may have already become apparent is that we shall be talking about behaviour rather than knowledge (as is more usual in AI). There is not time to discuss this here but some indication of the reasons behind it can be found in (Wavish and Connah 1990, Maes 1993a).

I will start to describe some of the implications which the theory has for the design of agents by discussing, in the next section, the particular kind of representation that is a common feature of situated agents.

3.0. Representation

Representation has always been a central issue in AI. If you see intelligence as being about problem solving and if your strategy for obtaining a solution is the use of formal methods operating on an abstraction from the problem in question then much hangs on choosing the right representation. An interesting early example of the importance attached to representation in this context can be seen in (Amarel 1968).

There are problems with creating such an internal representation. First there are the overheads associated with the construction of the model. Then there is the difficulty of ensuring that the model is always up-to-date. This is particularly difficult if there are other agents in the world causing it to change in unpredictable ways. (It is a remarkable fact that for many years AI was concerned only with situations where there was only one agent in the world). Finally, given that the model is properly maintained, there is the computer power necessary to cope with searching, inferencing and planning and this bring us up against not only the practical problems of resource limitations but also the more serious problems of tractability in principle (Chapman 1987).

Is there an alternative kind of representation that doesn't come up against these problems? The answer provided in the situated agent paradigm is to say that the world is the best representation of itself (Brooks 1991). In this case there are no overheads for model construction and maintenance, and the model is always accurate as well as being as rich as only the real world can be. But there is more to it than this. When an internal model has to be constructed there is no way in which the agent can know at the time of construction which parts of the model are going to be of importance and which not. Indeed as this is likely to change with time, any information that could conceivably be required must be embodied in the model. The model is very 'flat'; no part is intrinsically more important than any other part. If,

for example, the agent is in a lecture theatre containing many chairs, each chair will have to be individually represented at all times in as much detail as could be required at any time. In the 'no-internal-representation' case the agent need only, at any given moment, perceive that part of the world that is currently of interest. In other words there is a *focus of attention.*

Let me try to clarify some of these ideas with the help of an illustration[1].

The task at hand is a task in the blocks world. The agent is presented with a table containing piles ('columns') of blocks. Each block has a letter on its front face. The task of the agent is to use the blocks on the table to create a copy of the rightmost column on the table. The world in which the agent has to perform its task is shown in Figure 1.

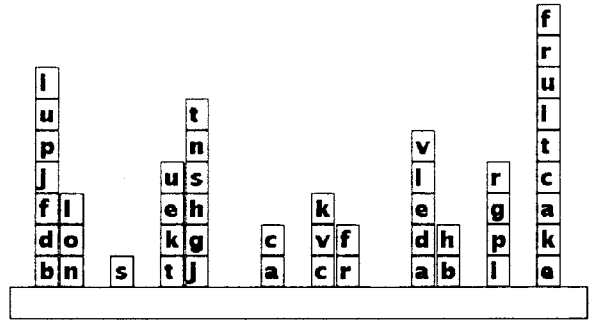


Figure 1.

Faced with such a task an AI programmer would typically try to deal with it in terms of planning (Rich 1983, p.259, Nilsson 1980, p.275). This would involve creating a representation of the world in the form of symbolic descriptions and a set of operators to act on those symbolic descriptions. There would also be an explicit symbolic description of the goal of the program or agent and some reasoning process by which the agent would discover how to change the current state of the world into the goal state. The planning paradigm also requires some method of detecting when a solution has been found and, in some cases, a method of recognising when a solution is nearly correct (Rich 1983).

A partial description of the above world in the style of a planning program might look something like:

ON(c,a)

CLEAR(t)

---

1. Chapman in (Chapman 1991) said "I believe that implementations in AI most often serve as *illustrations* of approaches" and "Approaches are learned in part from the practice of reimplementing illustrative programs". The work described here is taken from my re-implementation and extension of Chapman's Blockhead program (Chapman 1989).

...

ONTABLE(s)

There would also be a robot arm which could be used to pick blocks up, put them down, and stack or unstack them. The arm (hand) can be empty so we can have states and operators such as:

ARMEMPTY

PICKUP(k)

STACK(v,h)

...

and so on.

The goal state might be described as:

CLEAR(f) ∧ ON(f,r) ∧ ON(r,u) ∧...∧ ON(k,e) ∧ ONTABLE(e)

(All of this ignores that, as here, there might be more than one block marked with a particular letter).

A simple plan is then a list of operators taking the original state into the goal state step by step.

The situated action approach to the same task differs in every respect from that of planning. There is no symbolic description of the world; there is only the world itself. There is no reasoning process[2] but only actions which are appropriate to the situation of the agent. In the current implementation at least, there is no explicit goal; the designer has an explicit goal but the goal-like behaviour of the agent is emergent from its atomic situated actions. (In the situated action paradigm, there is an important distinction to be made between the point of view of the designer or observer and that of the agent. This distinction is often blurred or non-existent in planning agents). Situated action rules may appear superficially to be similar to operators with pre-conditions and post-conditions but whereas the latter work on the internal representation, the situated actions of the agent form part of a loop linking the agent directly into the rest of the world via perception and action

Perception in this example is simulated by intermediate level vision. This simulation is accomplished by means of markers (Chapman 1989, Chapman 1991, Ullman 1984). A marker is a device within the visual system of the agent which can be used to (detect and) mark specific features in its visual field. For example the bottom or the top of a column of blocks. It can also be used to track some ongoing activity such as marking the place where the next block is to be placed. This kind of marker merely marks a position in the visual field of the agent. It is the

2. In a sense there is a reasoning process but it is 'built-in' to the actions which are determined by the situation. This kind of implicit reasoning is a common feature of skilled activity.

responsibility of the agent to get the marker to the right location in the first place. Certain markers can also return directly the identity of a block (i.e. the letter on its front face) as was mentioned above. Markers are the only visual interaction that the agent has with the world. In other words if there is no marker at any given time on a particular spot the agent can have no knowledge (at that time) of what is there or even of the existence of the location[3]. If something happens in a location where there is no marker the agent will be totally unaware of it. This, in effect, constitutes a focus of attention for the agent and once again it is the responsibility of the agent to ensure that its markers are handled in such a way that it is always attending to those things that require its attention if it is to perform its task satisfactorily. The manipulation of markers is accomplished by the situated actions of the agent.

By manipulating a small set of markers it is possible to find blocks, decide which ones should be moved and subsequently keep track of them in such a way that a copy of the right hand column of bricks can be made. It is not a difficult matter to extend the program so that if the construction of the copy is interrupted either maliciously or by some 'natural' event like the pile collapsing, the agent can still recover from the situation using exactly the kinds of actions it used in constructing the pile in the first place. This kind of recovery is difficult or impossible when using an internal representation but then it is never seen as necessary since nothing can interfere in the only world it knows: the internal representation itself. This is why the blocks world, as usually described, is so unrealistic.

3.1. Summary

Here is a brief summary of the advantage of dispensing with internal representations.

- the representation does not need to be updated. The world is never out of date.
- no representation of the world can be as rich as the world itself. The accessibility of this rich resource is limited only by the perceptual abilities of the agent. In particular temporal and spatial relationships are automatically maintained in the world.
- the representation is automatically shared between agents and can form the foundation of communications between them.

This is by no means the full story on representations; in particular, I am not suggesting that agents never use internal representations. I am simply saying that for many of their activities internal representations are not only not necessary but may be counter-productive. I shall have a final comment to make about representations in the section on the future of situated agents.

---

3. Note that markers can also be used to mark 'virtual' objects such as the gap between two objects or the space into which something has to be placed.

4.0. The Social Nature of Agents

It was remarked earlier that much of AI has been concerned with the operation of a single agent in a passive world. The apparent assumption has been that once one agent had been successfully designed it would simply be a question of adding other rather similar ones.

Although one can think of applications where an agent acts in isolation, the majority of cases, and particularly those falling under the rubric of this Summer School, involve cooperation between agents. Even in cases of apparent isolation (e.g. a Mars rover) the behaviour of the agent will have been learned or given to it, and its concepts will have been acquired, in a multi-agent society. Without that society its actions would have no purpose and it would not have formulated the concepts which appear to drive it. In other words, the science of autonomous agents is, in an important sense, a social science. This idea of a society of agents is an extension of the idea of situated agents where account is taken of the fact that the agent is not merely situated in the physical world but specifically in a world where there are other agents with which it has to interact.

The behaviour of real people in a society consists for the most part of complementary activities: questions require answers, one kind of gesture elicits another, handing over an object triggers the taking of the object and so on. Such actions only have any significance in a social context. When we are designing agents which are going to interact with one another we have to bear this in mind. We have to design the actions of the agents so that they can engage in this kind of mutually supportive activity.

An example of a specifically social activity is provided by cooperation. Here there is an implicit communication between agents which is effected by the actions of the agents in the world. Of course in one sense *all* communication travels via the world (sound, light etc.) but I am referring to something different in this case. Suppose two bricklayers are building a wall. As one of them lays a brick in place the shape of the wall is changed and the other bricklayer is guided in his placement of the brick by the new shape of the wall. This is a very simple example but other examples can be quoted. For example the cooperation between two agents assembling something can be achieved in a similar way (Hickman and Shiels 1990). All that is required is the ability of the agents to perceive the current state of the wall or the assembly and to have actions which are appropriate to that state. The significance of each action is dependent on the state of the wall or assembly.

I will illustrate this process with examples suggested by a demonstration program (Wavish 1991). This program is in the nature of a game or entertainment based on the idea of a sheep dog rounding up sheep and driving them into a pen. The demonstration is a simulation of a system containing seven agents: a sheep dog, five sheep and a shepherd. The idea is that the dog should round up the sheep and drive them towards the shepherd. For this to happen both the sheep and the dog must, in some simple way, be able to perceive each other. As far as the dog is concerned, its

surroundings are notionally divided into eight sectors centred on the dog itself. It has a rudimentary 'vision' system which informs it of which octants contain sheep. (This is all; it doesn't know how many sheep or how far away they are). The sheep on the other hand are 'aware' of the presence of the dog when it gets too close for comfort. Given this rather basic perceptual ability, what behaviour should we write down that will cause the dog to round up the sheep and herd them towards the pen? The following four behaviours are sufficient:

- if the dog is running along with the sheep on its right hand side and there are no sheep in the dog's front-right octant, it should turn right until sheep appear in that octant (similarly if it is running with the sheep on its left).
- if, at any time, the shepherd appears in one of the dog's three front octants, it must reverse its direction of travel.
- if the dog approaches a sheep too closely the sheep must move (run) away from the dog.
- the sheep must avoid colliding with each other.

The result of these four behaviours is that the sheep are compressed into a flock and that the dog weaves to-and-fro behind the flock, driving it towards the shepherd. This behaviour can be seen in Figure 2.

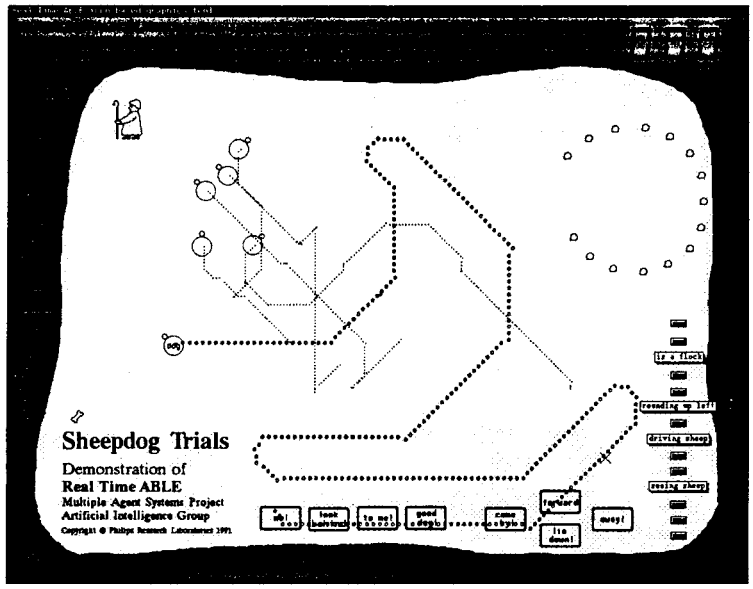


Figure 2.

Ignoring the details of the diagram, one can clearly see the tracks of the sheep and of the dog as it weaves to-and-fro.

What points are illustrated by this fragment of the demonstration?

1) The behaviour of the agents is situated. In this simple example it is easy to

write down, for example, what conditions must hold for the sheep to run away from the dog.

2) The four behaviours described are (indeed have to be) present concurrently. It makes no sense to think in terms of priorities or sequences for these behaviours; any or all of them must be triggered in the appropriate situation.

3) The flocking and herding of the sheep is not achieved solely by the dog's behaviour as one might assume but requires the complementary behaviour of the sheep. It is, in this sense, a kind of social behaviour in which all the agents (dog and five sheep) trigger behaviour in each other leading to stable behaviour of the social group as a whole.

The last point really contains the essence of the complementary nature of social behaviour and so it is worth emphasising. An analysis of what is happening when a sheepdog is used to round up sheep might suggest that the dog is forming the sheep into a small group (a flock) and then driving the flock towards the shepherd. However, this analytical statement is couched in terms which are appropriate to the observer of the scene; they are, as Maturana would say (Maturana and Varela 1970), descriptive terms in the observer's domain. In this instance, because we know exactly how the program has been written, we can say categorically that this description is misleading. There is no sense in which the dog is actively forming a flock and the sheep are passively allowing themselves to be herded in this way or that the dog is *driving* the sheep towards the shepherd. It appears this way because it is *the purpose of the designer* that this should happen and thus it is described by the observer in these terms. The situation is much more symmetrical between the dog and the sheep. All that one can say objectively is that the dog has one set of behaviours and the sheep another and that the result of the interaction of these behaviours in the system as a whole is that dog and sheep move in a way which satisfies the designer's purpose.

Notice that no mention has been made of the goals of the dog or the sheep or of their beliefs or desires or, indeed, of their plans. These concepts are unnecessary for the implementation of the various agents in the scenario. A belief, in this context, is not an explicit sentence or state in an agent but a concept used by an observer to make sense of the behaviour of the agent from the observer's point of view. Of course it is possible to verbalise plans, goals and beliefs but this is simply the agent observing its own behaviour and even then they may not fulfil the role assigned to them in most AI literature. Plans, for example, as (Suchman,1987) puts it, are just one resource for action amongst many. Except in the simplest cases they can not be a list of actions to be undertaken sequentially in order to achieve a pre-determined goal.

5.0. Emergent Behaviour

It has been observed by a number of people (Wittgenstein 1953, Dreyfus 1992) that

complex human behaviour cannot be explained by supposing that the person is (even unconsciously) obeying a fixed set of rules. There are difficulties, for example, in the handling of exceptions or in the flexible interpretation of the rules. The latter problem can lead to an infinite regress because of the need to have rules for the interpretation which are themselves subject to interpretation.

I do not know if there is a 'solution' to this problem, but there is an approach which I think is worth pursuing. When watching the demonstration of the sheep and sheepdog game, observers find it is natural to describe it in terms of words or phrases such as 'flock', 'rounding up', 'herding the flock' and so on. None of these concepts is explicitly represented in the program but they are ascribed by the observer to the behaviours of agents in the system. Furthermore the apparent behaviour that they are describing is, as was pointed out earlier, behaviour which is emergent from the interaction of other behaviours such as running towards the sheep or running away from the dog. I do not know whether this kind of behaviour is what others mean by emergent behaviour - there seems to be a considerable degree of confusion as to what it is or, indeed, as to whether it exists at all - but whatever it is called it is of considerable interest for two reasons. In the first place it seems to be a way of achieving a desired behaviour e.g. rounding up sheep without explicitly programming it and in the second place such behaviours (the 'flock') can be very stable.

Going back for a moment to the non-explicit way in which this is achieved, I think that trying to get this kind of stable behaviour by explicit means would have been very difficult. One would have needed some kind of definition of flock which would have come up against precisely the kind of objections made by those who object to rule-following: if its size were slightly larger than that proposed in the definition it would technically not be a flock any more or, if the shape were not sufficiently circular (say), it could no longer be called a flock within the program and so on. When we ascribe behaviour these problems don't exist. Of course we are still using rules but these rules are, as it were, at a different and simpler level; they are rules about the motion and the perceptual acts of the dog and sheep and not about such abstract things as rounding up. The rules at this level can be quite rigid without the emergent behaviour to which they give rise appearing to be rigid also[4].

The stability of the emergent behaviour in this case is important. The overall behaviour of the group of agents is not predictable in detail (although it is deterministic) and different starting positions of the agents lead to sequences which, in detail, are completely different. However the behaviour at the level of the flock (and this indeed is what leads us to *ascribe* the term 'flock') is extremely stable and robust; it can be deliberately disrupted (e.g. the player can pick up and move the dog or any or all of the sheep) and will fairly rapidly return to the stable flocking and driving behaviour. This combination of stability with lack of any detailed

4. One can imagine some of these lower level behaviours, in their turn, being emergent in the above sense. I am ignoring that here for the sake of simplicity.

predictability is characteristic of a chaotic system and we might speculate that 'flock' is the name we give to an attractor in such a system. We have not as yet pursued this idea any further. The applicability of chaos theory to agents has been discussed by Kiss in (Kiss 1991).

6.0. Skill

The idea of skilled agents has already figured in this lecture. In fact one of the changes in viewpoint that I would like to argue for is that agents should primarily be seen as skilled rather than intelligent. There are at least three reasons for this. The first is that skill is in many ways a more straightforward concept than intelligence. There have been endless debates about what constitutes intelligence and how it should be measured. Skill on the other hand is not so controversial, at least at the level of simple physical skills. We all recognise it and it is perfectly possible to acquire skills by well-defined methods of training. Measurement of this type of skill in particular instances is also relatively straightforward. A second reason for preferring the idea of skilled rather than intelligent agents is that for many, perhaps most, of the things we want agents to do in real applications, skill is what is required. This fact has been obscured by the tendency in the past for AI to concentrate its attention on those highly intellectual pursuits that form only a tiny proportion of human activity. AI has always had difficulty with common sense, for example, because it has tried the approach of trying to formalise common sense knowledge and then treating it in a rational way. In fact common sense is what we learn from our embodied existence in the world and in this it has much more in common with skill than with intelligence. Agents in the interface may not require great reasoning powers but rather skill at the job and some small degree of common sense to avoid gross and, to people, unintelligible mistakes. The final reason for concentrating on skill is that one can speculate that reasoning is based on skilled activities in the real world and is not something that is pursued by people in a completely abstract way (Chapman and Agre 1986).

Let me give just two examples to support this last speculation. Solving problems and puzzles of various kinds has always formed an important part of the subject matter of mainstream AI; missionaries and cannibals, monkeys and bananas not to mention the prisoner and his perennial dilemma all figure prominently in the literature. But what about something like a jigsaw puzzle? It seems to me fairly clear that an approach to doing a jigsaw puzzle which relied on reasoning with a formal representation of the shape of each piece together with a formal description of the fragment of the picture appearing on the piece would be a very difficult way of trying to solve the puzzle. This is clearly not what people do. As we discussed at some length in the section on representation, in this case the puzzle exists in the world and does not require a complete internal representation on the part of the agent. The point I am trying to make here however is that whatever reasoning takes place in solving the puzzle also requires the agent to be able to *see* the pieces, to *handle* them, and to *move* them, often actually trying them out before deciding

whether they fit or not. It is the strong physical element in the reasoning which I want to emphasise. The other example I wish to give concerns the solution of mathematical problems. Here, at first sight, we have the prototypical intellectual activity. Surely abstraction and formality are the very essence of mathematics. And yet when one is actually doing mathematics there is a strong element of *manipulation*; terms are grouped together, *moved* from one side of an equation to the other or *rearranged* in other ways which are as much a product of skill and experience as of abstract reason. We talk about *seeing* a solution and a computer program which has no notion of *seeing*, *touching*, or *moving* will, I submit, be considerably handicapped when trying to tackle this kind of problem.

Wavish (Wavish 1993) has written a program which demonstrates precisely these kinds of skills in an agent. His agent plays the game of Tetris. In this game (if you are not familiar with it) blocks of various shapes appear to fall under gravity and land either on the 'ground' or on top of blocks which have previously fallen down. The aim of the player is to manipulate these blocks as they are falling either by rotating them or by moving them sideways (translating them) in such a way as to fit them into holes in the already existing heap of blocks. If the player does this successfully (by which is meant by completely filling rows of blocks) then he or she scores points. Additionally, when a complete row is achieved, the row is deleted and the general level of the heap goes down. It is an important part of the strategy to keep the height of the heap down since otherwise it reaches a level at which the game is terminated. The game is shown in progress in Figure 3.

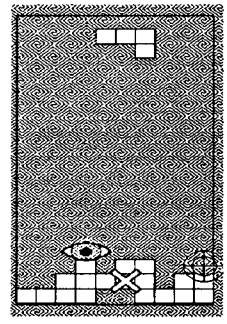


Figure 3.

(The other icons visible in the diagram represent the various markers that the agent uses in playing the game. See the section on representation). The agent plays the game in a situated manner. It scans the heap (hence the 'eye' icon) looking for certain patterns in its top surface and it then applies rotations and translations to the block depending on which block is falling and which patterns it finds. It is important to realise that within the agent there is no representation of the heap of blocks; the agent plays rather as one might do the jigsaw puzzle described above. In fact the agent does not scan the falling block also but this would be an obvious extension of its behaviour. The game is actually rather more complex than I suggested above since one gets more points if two, three or four rows are completed simultaneously.

Thus it sometimes pays to defer an obvious move in the hope that a later block will complete several rows.

The impression that one gets on watching the agent play is of some (fairly small) degree of intelligence but in fact nothing that we would normally think of as intelligence is involved. Most of the time, although the agent is by no means optimised, it seems to choose a reasonable location and orientation for the block. At all events the scores it achieves are roughly speaking an order of magnitude better than my best score. I would not like to think that this was a true reflection of our relative intelligences although I can bear to think that it is more skilful than I am.

7.0. The Future of Situated agents

Research on situated agents has not been going on for very long. The theory is still patchy and there are few commercial applications extant. Nevertheless, it is an area which is developing fast and which seems to have great potential. An attempt has been made in this lecture to introduce the topic by discussing some of the important issues but how will the theory develop and what kinds of application can we expect in the future?

Probably the hottest topic at present is that of learning in situated agents. The Tetris agent described in the previous section was hand-crafted; the various combinations of block shape and heap patterns were chosen and refined by the programmer. Whilst this is feasible for small programs it would constitute a bottle-neck in the writing of larger ones. For this reason it has been suggested by several people that it will be necessary for agents of this kind to be learning agents. Work in this direction has been done by Maes (Maes 1991, Maes 1993b) and Chapman (Chapman 1991). This work is still in its early stages and Chapman, in particular, has reported that his attempts to harness backpropagation methods has been disappointing. He concluded that in future it would be necessary (because of the situatedness of the agents) to look at learning methods which are much more directly linked to the ways in which the perceptions and actions of the agent are effected.

As well as being practically motivated there is a strong theoretical interest in situated agents which can learn. It is typical of the situated approach that it engenders *holistic* solutions; it seems very likely that a proper treatment of learning will be impossible without simultaneously developing theories on perception and memory (Chapman 1991, Edelman 1987). This is in clear contrast to the mainstream AI approach which would (indeed, has) separated out these functions and treated them independently. This is a difficult task. However not only does the situated approach require such holism, it also makes it possible by virtue of the way in which agents are embedded in their environments. I expect this area to be one of the most exciting areas of research in the next few years.

The discussion of representation in the lecture was confined to those situations in

which the world of the agent could be used as the only representation required by the agent. This was an important insight made in some of the early work on situated agents (Brooks 1986, Agre & Chapman 1987). However it is unlikely to be the whole story; the issue of representation goes beyond this. There is, for example, the particularly interesting situation in which dynamic external representations (e.g. speech, drawings, writing) are used by an agent as part of its current situation. As Clancey says (Clancey 1991): *"We don't know what we want to say until we say it."* There is an enormous amount of work to be done on this question of representation; the most that can be said of work to date is that it has challenged the accepted view about internal representations and opened up the whole field to a radical re-appraisal.

There are still, as it were, two schools of thought in AI. Mainstream workers are still interested in the problem solving aspects of the subject whilst the situated agent people see things much more in terms of skilled behaviour. There have been no successful attempts to bridge the gap. In fact, the gap is rather large and therefore daunting but sooner or later someone is going to have to try to cross it.

The topic of natural language has been conspicuous by its absence from the body of the lecture. The principal reason is that whatever work has been proceeding on situated language, very little of it has filtered through to the situated agent community and yet I believe that ultimately this must happen for two reasons: first is the intrinsic importance of natural language to the cognition of agents and second is the revitalising thrust which I believe a holistic approach would give to the field. The only work along these lines of which I am aware is described by Ward (Ward 1992).

In the section on the social behaviour of agents I emphasised the complementary nature of the behaviour of agents. This is really just a hint of a much more radical position which I think will eventually have to be embraced in this paradigm although it is not clear at present how this can be done. I am referring to the idea that cognition and intelligence, if you will, are not things that are contained within the head of the agent but are products of the social interaction of agents (Coulter 1979, Costall 1991). If this line of argument holds up it is fruitless to continue to try to make agents more intelligent simply by putting more and more 'clever' things into them. They will only behave in ways which we would remotely call intelligent if they have a rich interaction with each other and with the world. It is because of this requirement for a rich and close interaction that I feel that situated agents are perhaps our best bet at the moment.

Finally applications. I am not aware of any genuinely commercial applications of situated agents that have been made so far. There have, on the other hand, been quite a lot of projects which have demonstrated their potential. It seems likely that the man-machine interface will be one area where applications will appear before long. In the slightly longer term multi-media, virtual reality and interactive drama (Bates 1990) are all areas where skilled agents could be expected to play important roles. Agents are being introduced into educational systems: work is proceeding in this

area at IPO (van Hoe and Masthoff 1993). They can also be expected to appear in training scenarios and command and control training exercises. In short there is no lack of potential applications and the technology is maturing rapidly. We can expect the results of the fusion of the two in the fairly near future.

References

Agre, P.E. (1995) "Routines", MIT AI Memo 828.

Amarel, Saul (1968) "On Representations of Problems of Reasoning about Actions", Machine Intelligence 3 (ed. Michie), Edinburgh UP, (1968) 131-171.

Bates, Joseph (1990) "Computational Drama in Oz", Working notes of the AAAI Workshop on Interactive Fiction and Synthetic Realities, Boston, July 1990.

Agre, P.E and Chapman, D. (1987) "Pengi: An Implementation of a Theory of Activity", Proceedings of the AAAI Conference, Seattle, Washington.

Brooks, R.A. (1986) "A Robust Layered Control System For a mobile Robot", IEEE Journal of Robotics and Automation, vol. RA-2, No. 1, March 1986, 14-23.

Brooks, R.A. (1991) "Intelligence without representation", Artificial Intelligence 47 (1991) 139-159

Chapman, D. (1987) "Planning for Conjunctive Goals", Artificial Intelligence 32, (1987), 333-377.

Chapman, D. (1989) "Penguins Can Make Cake", AI Magazine, Winter 1989, pp.45-50.

Chapman, David (1991) "Vision, Instruction, and Action", MIT Press, Cambridge MA, London, England

Chapman, D and Agre, P.E. (1986) "Abstract Reasoning as Emergent from Concrete Activity" in "Workshop on Reasoning about Actions and Plans", ed. Georgeff and Lansky, Timberline, Oregon, publ. Morgan Kaufmann.

Clancey, W.J. (1991) "Israel Rosenfield, The Invention of Memory: A New View of the Brain", Artificial Intelligence, 50 (1991) 241-284.

Costall, A. (1991) "Graceful Degradation", in "Against Cognitivism: Alternative Foundations for Cognitive Psychology", Arthur Still and Alan Costall (Eds., London, Harvester Wheatsheaf (1991).)

Coulter, J. (1979) "The Social Construction of Mind: Studies in Ethnomethodology & Linguistic Philosophy", publ. Macmillan Press.

Dreyfus, H.L. (1992) "What Computers Still Can't Do: a Critique of Artificial reason", Cambridge, MIT Press 1992.

Edelman, G.M. (1987) "Neural Darwinism: The Theory of Neuronal group Selection", New York, Basic Books, 1987.

Hickman, S.J. and Shiels, M.A. (1990) "Situated Action as a Basis for Cooperation", in Proceedings of the 2nd. European Workshop on Modelling an Autonomous Agent in a Multi-Agent World, Paris, 1990, publ. Elsevier-North Holland.

van Hoe, R.R.G and Masthoff, J.F.M. (1993) "A multi-agent approach to interactive learning environments" in 'Collaborative problem solving: theoretical frameworks and innovative systems' AI-ED 93 Workshop, Edinburgh, Scotland.

Kiss, G. (1991) "Autonomous Agents, AI and Chaos Theory", in "From Animals to Animats", eds. Meyer and Wilson, MIT Press.

Laurel, B. (1991) "Computers as Theatre", Addison-Wesley, 1991.

Maes, P. (1991) "Learning Behaviour networks from Experience", Proceedings of the First European Conference on Artificial Life, MIT Press, 1991.

Maes, P. (1993a) "Behavior-Based Artificial Intelligence", Proceedings of the 2nd. Conference on Adaptive behavior, MIT Press, 1993.

Maes, P. (1993b) "A Learning Interface Agent for Scheduling Meetings", Proceedings ACM - SIGCHI, Florida, 1993.

Maturana, H.R. and Varela, F.J. (1970) "Autopoiesis and Cognition: The Realization of the Living", D. Reidel Publ. Co. Dordrecht, Holland and London England, 1970.

Nilsson, N.J. (1980) "Principles of Artificial Intelligence", Tioga Publishing Co.

Rich, Elaine (1983), "Artificial Intelligence", McGraw Hill.

Suchman, L. (1987) "Plans and Situated Actions: the problem of human-machine communication", Cambridge University Press.

Ullman, S. (1984) "Visual Routines", Cognition, 18, pp. 97-159.

Ward, N. (1992) "A Parallel Approach to Syntax for Generation", Artificial Intelligence 57 (1992) 183-225.

Wavish, P.R. (1991) "Exploiting Emergent Behaviour in Multi-Agent Systems", Proceedings of the 3rd. European Workshop on Modelling an Autonomous Agent in a Multi-Agent World", Kaiserslautern, 5-7 August 1991, eds. Y. Demazeau and E. Werner, Elsevier Science Publishers B.V. (North Holland), 1991.

Wavish, P.R. and Connah, D.M. (1990) "Representing Multi-Agent Worlds in ABLE", Philips Research Laboratories, Redhill, England, Technical Note Number 2964.

Wavish, P.R. (1993) Private Communication.

Wittgenstein, L. (1953) "Philosophical Investigations" Blackwell.

Referring in a Shared Workspace

Anita H.M. Cremers

Institute for Perception Research, P.O. Box 513, 5600 MB Eindhoven, The Netherlands

Abstract. In this paper referring acts to objects (expressions as well as gestures) that were used in dialogues during a block building experiment in a shared workspace are described. A classification of information types that were used in the referring expressions is given, which consists of four main categories, i.e. reference to physical features of the object, reference to the location of the object in the domain, reference to the orientation of the object in the domain, and reference to the so-called history of the object. Also the influence of mutual knowledge of the participants (either knowledge about the dialogue or about the domain) on the used referring acts is described. The focus of attention turns out to play an important role in the use of references. It is argued that, beside the effect of dialogue focus on referring acts, especially the effect of the domain focus needs to be investigated further.

Keywords. Referring expressions, referring gestures, dialogues, mutual knowledge, focus of attention

1 Introduction

When two people are talking about a task they should perform together, they will have to indicate to eachother which of the available materials or tools they are going to use. So, in this type of situation they will use a lot of *referring acts*, i.e. verbal and non-verbal means to single out an object in space, so that the partner will be able to locate it.

Research on referring behaviour to objects or places in the extra-linguistic context has mainly focused on the linguistic part of the referential act. Spatial conception can be encoded in linguistic structures in many ways, for instance, by means of deixis, relative locations or motions, absolute systems and place names or descriptions (Levinson 1992). The decision of a speaker to utter a particular expression to refer to an object largely depends on the type of coordinate system that is used, the place of the origin of this system and, possibly, the chosen relatum, i.e. the object or person that is chosen as a reference point with respect to which this object is located (Levelt 1989). However, in face-to-face conversations, in many cases these utterances are accompanied by non-verbal communicative acts, such as gestures, facial

expressions and/or bodily orientations. Especially (pointing) gestures are of vital importance for the hearer to be able to identify the right object or place, since they can provide a very precise indication of their location. Although much is already known about linguistic means of reference, it has still not been investigated enough how people refer, verbally as well as by means of gestures, to objects or places in an actual dialogue situation.

The goal of the present research is to investigate how the speaker's assumption of the hearer's knowledge influences the information that is included in the referential acts. This research is based on the hypothesis that form ('what information is included in the expression?') and content ('which object does the expression refer to?') of referential acts strongly depend on the mutual knowledge of speaker and hearer (the *common ground*, as Clark and Wilkes-Gibbs call it (Clark and Wilkes-Gibbs 1986)). This can either be general knowledge about the domain of discourse (e.g. properties of objects in the domain), knowledge about the actual state and history of the domain of discourse (e.g. the past or the present location of a certain object) or knowledge about the preceding utterances in the dialogue (e.g. a name that has been given to a certain object).

In this paper, first the methodology of a dialogue experiment, that was carried out to study referring expressions to objects in a relatively restricted domain, will be described. Then, an overview will be given of the types and amounts of referring expressions that were used in the dialogues. Finally, the influence of the speaker's and the hearer's knowledge on the type of referential act used will be outlined.

2 Methodology

In order to test the hypothesis that was mentioned in the introduction a dialogue experiment was carried out in which 10 pairs of Dutch subjects participated. The experimental set-up and task were designed as to evoke as many varied referential acts as possible. One of the dialogue partners (A) was told to instruct the other partner (B) in detail to first build, and second, rebuild a block-building on a toy foundation plate on the basis of a provided example. The block-building consisted of blocks in four different colours, three sizes and four shapes.The partners were seated side by side at a table, but were separated by a screen. Only their hands were visible to each other, provided that these were placed on top of the table in the neighbourhood of the foundation plate. All subjects were allowed to observe the building domain, to talk about it, and to gesticulate in it, but only the respective builders were allwed to manipulate blocks. The set-up of the experiment is depicted in figure 1. The resulting 20 building sessions, of which the dialogues were much like Grosz's task dialogues (Grosz 1977), were recorded on video-tape.

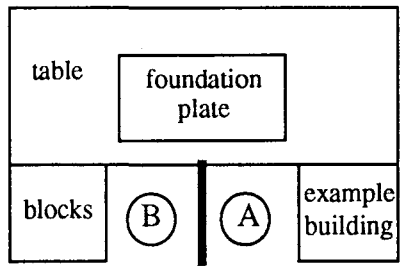


Fig. 1. Experimental set-up (top view)

3 Referring expressions in the dialogues

In the dialogues four main categories of referential means could be distinguished, namely reference to physical features of the object, reference to the location of the object in the domain, reference to the orientation of the object in the domain and, finally, reference to the so-called history of the object which was developed in the course of actions that were carried out in the domain and the topics that were discussed in the dialogue. These four categories will be discussed in the subsections 3.1 through 3.4 below. In this paper only the results of the re-building sessions are included for analysis, since the referring expressions in these 10 dialogues showed more variation, since the participants had to carry out more varied actions. In the selected dialogues a total amount of 665 referring expressions occurred, of which 59.5% (396) actually contained information out of one or more of the four categories mentioned above. These expressions either acted as direct references to objects in the domain (62.4% of them, which means an amount of 247) or as anaphora (37.6%, 149). The remaining 40.5% (269) only consisted of pronomina or demonstratives.The four categories of information-containing references will be discussed below.

3.1 Reference to Physical Features of an Object

By far the majority of the referring expressions that were used included information about physical features of the object that had to be identified. In fact 92.2% (365) of the referring expressions included this type of information, and 77.5% (307) of the expressions consisted of this information type only. The physical properties that were used in the dialogues were colour (e.g. *red*[1]), size (e.g. *large*) and shape (e.g. *square*) of the blocks, which actually were the three distinguishing features that

1. Since the examples of referential expressions provided in this paper are merely abstractions of the Dutch expressions that were used in the dialogues, they are only given in English.

were present in the set of blocks available. There was a feature preference for colour, since this was used in 97.3% (355) of the references that contained information about physical features, whereas size and shape were mentioned in, respectively, only 25.2% (92) and 17.8% (65) of the cases.

3.2 Reference to the Location of an Object

Reference to the location of an object was mainly used by dialogue participants who did not use pointing gestures at all. This definitely makes sense, because both referential means seem to accomplish the same effect, one by means of language, the other through gesturing.

1. Location in general
Dialogue participants sometimes used general referring expressions to objects in the domain, e.g. by means of *here* and *there*. In 3.8% (15) of the total amount of information-containing referring expressions used this was done, and in 1.8% (7) of the cases this reference type was used in isolation. When these references were used to refer directly (non-anaphorically) to objects in the domain pointing gestures were always added. Most of the occurrences of this type of reference occurred at points where the speaker shifted his or her attention to another region of the domain, for instance to give building instructions about a new part of the building. We shall call this kind of shift a transition of the *focus of attention*.

2. Location with respect to the participants
The location of an object was sometimes indicated by stating its relative position with respect to both participants, e.g. in *the blue one on the right*. This information type was used in 6.1% of the total amount of references and in 1.5% of the cases it was the only type. Since the participants were seated side by side, their perspectives towards the domain were almost the same, and no distinction between hearer- and speaker-based reference had to be drawn. These expressions were hardly ever accompanied by pointing gestures, probably because they already contained enough information to identify the intended object. Reference by means of indicating the location of an object with respect to the participants was again mainly used to install a new focus of attention in the domain.

3. Location with respect to (an)other object(s)
The location of an object was also indicated by means of stating its position with respect to another object or other objects in the domain, e.g. in *the blue block behind the yellow one*. Of course, in these expressions the position of the participants should also be taken into account. These expressions were used in 3.5% (14) of the cases, and in 0.8% (3) of the cases this was the only information type supplied. Reference to a location of an object with respect to (an)other object(s) is mainly used when the current block is located in the neighbourhood of the block referred to previously. In those cases the previous block is used as the relatum of the current one. Since in these cases the current object is located close to the previous one, no focus

transition occurs, it is rather maintained. That could also be reason why no pointing gestures were being used to accomapny this type of expression, since the location was already more or less clear.

4. Location with respect to the hand of a participant within the domain
The occurrence of referring expressions using the hand of a participant as a relatum is probably a consequence of the specific set-up of the task. In this set-up the participants share the same perspective, and the only difference between each other's bodily positions they can actually observe is the position of their hands, which may change constantly. Instances of this type of referring expressions occurred in 1.3% (5) of the cases, and in 0.8% (3) of the cases it was the only disambiguating information offered. No gestures accompanied these expressions.

The position of the partner's hand was used in two ways. First, the speaker can inform the partner on where a block is located with respect to the location of his hand at the moment of the utterance, e.g. *the blue block to your right.* Second, the speaker can tell the partner in what direction his hand should move in order to reach the intended object. So, in this case, the location of the hand is presupposed, and instead of informing the partner about the location of the object, the action he or she has to carry out is supplied. This action can either be a default-action or an explicitly indicated one. In the case of a default-action the partner is already moving in the right direction and should only be encouraged to proceed doing so, e.g. in *a bit further.* In the latter case the explicit direction should be provided, like in *go to the left.*

3.3 Reference to the Orientation of an Object

The different ways in which an object can be positioned in the domain all result in different object orientations. Speakers can make use of an object's orientation to distinguish it from other (identical) objects, e.g. in *the horizontal yellow block.* In 1.0% (4) of the total amount of referring expressions participants made use of this disambiguating device, but the information was always accompanied by other types of information. No accompanying gestures were used.

3.4 Reference to the History of an Object in the Domain or in the Dialogue

Finally, an object can be disambiguated by means of reference to earlier events in which the object was involved, e.g. in *the red one you have just put down.* Also, when one of the participants has already talked about the object earlier in the dialogue this can be used for disambiguation, e.g. in *the red one you just mentioned.*

In 7.1% (28) of the total amount of referring expressions participants made use of historical aspects, in 2.8% (11) of the cases this was the only information they provided. References to the domain history and the dialogue history occurred equally often. Hardly any accompanying gestures were used. The only ones that were used referred to objects that had been talked about before, and were still

located in the domain.

4 Influence of Mutual Knowledge on Referring Expressions

During the course of a dialogue and of events in the domain of conversation, knowledge is built up about these issues. Later in the dialogue and in the building process, partners may make use of the knowledge they assume the other has available. In fact, the assumption of the presence of this knowledge may to some extent determine the type of information that is being used in the current referring expression, although it may be fully disambiguating in its own right. This will be discussed further in the sections 4.1 and 4.2.

Beside fully disambiguating referential expressions, dialogue participants also use reduced expressions, i.e. expressions that would not be fully disambiguating when used in isolation. A reduced expression may be used when the speaker implicitly assumes that the partner already knows or can easily get to know the missing part of the fully disambiguating information he should actually have provided. In the sections 4.1.1 and 4.2.1 occurrences of reduced information will be discussed.

4.1 Influence of the Dialogue Context on the Type of Information

In the dialogues it could be demonstrated that participants made use of the contents of the preceding dialogue when uttering the current referring expression.

In the first place, when participants refer to the history of the dialogue (see section 3.4), they assume that there is mutual knowledge about the fact that this particular topic has been talked about previously in the dialogue.

Second, participants may, either explicitly or impicitly, agree on a name for a particular object, or a particular type of object. For example, in the dialogues a block with a deviant shape was agreed upon to be called *de glijbaan ('the slide')* by a couple of participants. Another couple agreed on calling the smallest block type *de kleine ('the small one')* in contrast to a larger type, which was called *de grote ('the large one')*, because there had been some misunderstanding earlier on how to tell the two types apart. Later in the dialogue the respective participants were able to use these names, when appropriate, without any problem.

Third, when one participant is busy trying to make clear which object he or she is referring to, but the other is not able to find out which object is hinted at by the description (e.g. *the large blue block*), the speaker will have to add some more information to help the hearer identifying it.[2] In the second 'turn' the speaker will usually use other types of information than the information types provided previously (e.g.

2. Note that this does not necessarily have to mean that the referential expression was ambiguous in itself, since the reason for the misunderstanding could also have been an inability on the part of the hearer.

the one on the right). So, in this case the speaker assumes that the hearer still knows the previous information, and provides only information that should be added to that knowledge. This means that, in general, given information does not have to be uttered for the second time, unless the hearer indicates that he or she has not understood what had been said.

Finally, if an object has been talked about recently, and the current block is located close to that block, then the previous block can be used as a relatum for describing the current block. In those cases a reference to the location of an object with respect to another object (see section 3.2) is often used. In fact, in 71.4% (10) of these expressions the relatum was the last object that had been mentioned before. For example in *place a large yellow block, place a small blue block to the right of the yellow one*, the large yellow block is used as a relatum for indicating the place of the small blue block.

4.1.1 Reduced Information as a Result of Dialogue Knowledge

Assumed knowledge about the dialogue on the part of the partner is reflected in the use of pronominal anaphora and ellipsis in the referring expressions. Anaphora can be used when the speaker assumes that the object referred to is in the focus of attention of both participants (Grosz 1977), in which case a pronominal reference suffices for making clear which object is meant (e.g. *take a small red block, put it on top of the large green one*). Ellipsis occurs when the referring expression in an utterance is omitted altogether, or when parts of the utterance are omitted. Total omissions may take place when both participants have already agreed upon which object is being referred to, and only some predication about the object is left to express, for example the destination of a particular block after having carried out the action on it (e.g. *take a small red block, put it on top of the large green one, (...) the green one on the right*). Partial omissions may occur when information in the current expression partially coincides with information in the preceding expression, e.g. in *place a small yellow block, put a blue one on top of it*. In this example the 'blue one' is taken to be a small block as well, although the explicit information is omitted.

4.2 Influence of the Domain Context on the Type of Information

In the dialogues, participants also made use of the knowledge about the history and the state of the domain of conversation they assumed their partner possessed. As was already discussed in section 3.4, participants can refer to events that have happened in the domain, like in *the block that just fell off*. Note that domain events can be referred to even when they have not been discussed earlier in the dialogue. Participants make use of the state of the domain by making sure that their referring act is as much disambiguating as possible within the domain. The speaker can do this by comparing the physical features of the block and the orientation of the block within the domain with features and orientations of the other blocks that are present. Also the position of the block with respect to the participants, with respect to other blocks

and with respect to the hand of a partner can be used to effectuate the disambiguation.

4.2.1 Reduced Information as a Result of Domain Knowledge: Focus of Attention

Knowledge about the domain of discourse and the manipulations that are or have been carried out in it may also give rise to the use of reduced expressions. Triggers for reduced reference within the domain that were observed in the experiment are the existence of either a spatial or a functional focus of attention.

1. Spatial focus
The spatial focus in the domain is the part of the domain that is being attended to more strongly than the rest of it. For objects that are located in a highly attended area the speaker does not have to provide fully disambiguating information in the used referring expressions. In these cases, the disambiguation should only concern the part of the domain that is attended to. The spatial focus can be created either explicitly verbally or implicitly or by means of (pointing) gestures.

- **Explicit**

The speaker may announce explicitly that objects that are located in a particular subdomain should be attended to more strongly, e.g. in *let's move to the right upper part*. In this way the speaker makes sure that the hearer focuses his attention on the indicated subdomain, so that he could use less information in expressions referring to objects that are located there. In the dialogues 6.1% of the total amount of referring expressions contained this type of information, and 1.5% of the same amount consisted exclusively of this information type.

- **Implicit**

The speaker may implicitly make use of the assumed focus of attention of the partner. The speaker can have well-founded reasons for believing that the partner's focus is actually directed to this particular sub-domain, for example when he has just referred to an object that is located there. In this case, the speaker might argue that the partner is inclined to consider the next reduced expression as a reference to an object in the neighbourhood of the one just mentioned. For example in *please remove these blocks (points at a green, a yellow and a blue block), the red one can remain seated there*, the speaker refers to a red block without fully disambiguating it. However, it is clear for the listener that the speaker refers to the red block in the neighbourhood of the three blocks that were pointed at earlier.

- **Gestures**

The partners may use their hands to point at objects, to indicate their orientation in the domain, to touch them, to pick them up or to hold them. Both the instructor and the builder are allowed to point at blocks and to touch them, but the latter can in addition pick up the blocks and hold them in his or her hand. Both partners can also make use of the fact that the other is (incidentally) pointing at or touching a block. In those cases the speaker can refer orally to that particular block by means of a minimally informative expression, like *that one*. The instructor can also use this mecha-

nism when the builder is picking up a block or holding it. In all of these cases the speaker can use less information than he would have needed if there had not been some involvement of a hand. In the dialogues participants used some kind of accompanying gesture in 16.8% of the total amount of referring expressions. However, three out of ten instructors did not use any gestures at all during the dialogues. Since the participants had been instructed to act as spontaneously as possible, this can just be considered a matter of preference. This lack of gesturing did not have a notable effect on the percentage mentioned above, because in these dialogues the builders had to use more gestures in order to verify whether they had correctly identified a particular block. This was less necessary for builders that had a pointing instructor as a partner.

2. Functional focus

Next to the assumed knowledge about spatial focus the speaker may make use of the partner's assumed knowledge about the current functional focus. Functional focus is related to the actions that have to be carried out in the domain. The concept of functional focus applies when the action that should be carried out more or less restricts the amount of blocks that can reasonably be involved in the action. In that case reduced information is possible, based on the assumed acquaintance of the partner with the preconditions- or post-conditions of the action.

- **Preconditions**

A partner can make use of the preconditions of an action when an object has to be removed. In that case, the object referred to is most likely the one that is located at a position that is easily reachable, e.g. in *remove the little yellow one*, which is taken to be the small yellow block on top of the building, although there are many other small yellow blocks available at positions that are not so easy to reach.

- **Post-conditions**

Post-conditions of an action can be used when an object is to be (re-)placed. Then the object referred to is most likely the one that fits best at the indicated location. For example, when there is an opening between two yellow blocks that can only contain a small block, the 'green block' in the utterance *place a green block between the yellow ones* is probably taken to be a small one.

In total the dialogues contain about 30 (12.1% of the total amount of information-containing direct references) cases of the implicit spatial focusing and the functional focus mechanism together. In these cases, it could be demonstrated that the partner was actually making use of one or both of these mechanisms, because the provided information was not fully distinguishing the referent object from the surrounding ones, and no pointing gestures were used. Actually, it was not always possible to distinguish between the use of implicit focusing or functional focusing, because these mechanisms coincided many times. For example, when a speaker has just been talking about a yellow block, and subsequently says that a large red block should be placed on top of a small green one, he probably means the green block that is placed in the neighbourhood of the yellow block, and is suitable for placing a large block on top of it.

5 Discussion and Conclusions

In this paper an analysis was given of referring acts (verbal as well as gestural) that occurred in 10 Dutch so-called task dialogues, and of the influence of mutual knowledge on the usage of these acts.

In the dialogues four main categories of referring expressions to objects occurred, namely reference to physical features, to the orientation or the location of objects, and to the history of an object in the domain or in the dialogue. Research on the naming of objects (Herrmann & Deutsch 1976, Herrmann 1983) has resulted in the finding that expressions used to single out an object in space depend on the objects surrounding the referent object. Speakers normally describe an object in such a way that it can be distinguished from other objects. In fact, these object naming are informative as well as non-redundant. Also, speakers tend to use the distinguishing feature that expresses the most pronounced difference between the referent object and the surrounding ones. In the dialogues the participants showed a strong preference for using physical features, especially colour. However, this may be due to the type of objects that were present in the domain and the task that had to be carried out.

It could be demonstrated that the participants made use of mutual knowledge about the contents of the preceding dialogue as well as the state and history of the domain of conversation when uttering the current referring expression. This was particularly obvious in cases where participants used reduced expressions, i.e. expressions that would not have been disambiguating if they had appeared in isolation. Reductions in information due to dialogue knowledge occurred in the form of pronominal anaphora and ellipsis. Reductions as a result of domain knowledge occurred especially when objects were positioned in the focus of attention of the participants, either as a result of their location and/or function or by means of hand gestures.

Knowledge about the *saliency* of a particular object in the domain of conversation may lead to the use of a reduced reference as well. Clark et al. (Clark, Schreuder and Buttrick 1983) showed that listeners select referents based on estimates of their mutual beliefs about perceptual salience, the speaker's goals, and the speaker's presuppositions and assertions. In our building domain saliency may relate to the physical features of an object, to the location of an object, to its orientation, or to its relative static or dynamic state. However, no clear examples of saliency effects could be found in the dialogues, probably because there were no objects with a clear inherent saliency present, and the task did not allow for using other saliency triggers.

The focus of attention appeared to be a central notion in the use and interpretation of referring expressions. At focus transitions participants tended to use more referring expressions to the location of an object in general and to the location with respect to the participants. Reference to the location of an object with respect to other objects was rather used when the focus did not shift (see section 3.2). Moreover, the focus of attention played an important role in the production and comprehension of reduced reference to objects. The effect of dialogue (or discourse) focus on the possible use of pronouns and ellipsis is well-known (Grosz 1977, Grosz 1981). However, the effects of domain focus, spatial as well

as functional and gesture-driven, deserve further research.

References

Clark, H.H., Schreuder, R., Buttrick, S. (1983) Common ground and the understanding of demonstrative reference. J. of Verbal Learning and Verbal Behavior 22, 245-258

Clark, H.H., Wilkes-Gibbs, D. (1986) Referring as a collaborative process. Cognition 22, 1-39

Grosz, B.J. (1977) The representation and use of focus in dialogue understanding. Technical Note 151. Menlo Park: SRI International

Grosz, B.J. (1981) Focusing and description in natural language dialogues. In: A. Joshi, B. Webber, I. Sag (eds.) Elements of discourse understanding. New York: Cambridge University Press

Herrmann, Th. (1983) Speech and situation: a psychological conception of situated speaking. Berlin: Springer

Herrmann, Th., Deutsch, W. (1976) Psychologie der Objektbenennung. Bern: Huber

Levelt, W.J.M. (1989) Speaking: from intention to articulation. Cambridge: MIT Press

Levinson, S.C. (1992) Primer for the field investigation of spatial description and conception. Pragmatics 2(1), 5-47

User Responses to Constraint Management in Computer-Mediated Design Activities: A Conceptual Framework

Donald L. Day[1]

[1] School of Information Studies, Syracuse University, Syracuse, NY 13244-4100, USA
D01DAYXX@SUVM.SYR.EDU (Internet)

Abstract. This paper discusses an ongoing study of user responses to constraints in tools used to design and generate computer software. The study also addresses the communication of design process rules and mental models from tool developers to tool users.[2]

Keywords. Constraint management, decision support, automated design, cognitive fit, computer-aided software engineering, process engineering.

1 Introduction

1.1 Relationship to Institute Themes

1.1.1 Human Learning, Perception and Cognition

This work proposes that spreading activation and semantic networks be used to characterize the decision-making process, and investigates user evaluation of relationships among constraints and design task options. It also examines user understanding of process modeling, and user perceptions of self within the work context.

1.1.2 Learning As An End-User Goal

Design tools examined in this study teach users appropriate procedures. Some tutor them about the software development principles underlying constraint implementation. Others inform individuals of the enterprise model followed by their organizations and alert them to quality assurance concerns.

1.1.3 Intelligent Support and Assistance To Users

Findings of studies such as this may be helpful in specifying constraint management systems that operate as knowledge bases. In such systems,

---

[2] Portions of this paper appeared originally as "Precis of Behavioral and Perceptual Responses to Constraint Management in Computer-Mediated Design Activities" in the *Electronic Journal of Communication/La Revue Electronique de Communication* ,Vol. 3, No. 2, April 1993.

automatically recorded transaction data describing constraint negotiation and user execution of design tasks may be used to create customized user-process profiles.

Sufficient understanding of the interplay between constraint implementation and user behavior also may help to develop intelligent assistants which can calibrate constraint flexibility dynamically, within task context, and project the impact of current decisions upon future choices and end-product characteristics. The result would be design tools that are as flexible as they can be without degrading product quality unacceptably.

1.1.4 Technologies, Tools and Techniques For Human-Machine Communication

The tools applied by users included in this study incorporate advanced knowledge representation, decision process and user interface design features which may affect responses to constraint management techniques. For example, graphical user interface tools and user interface management systems provide a wide range of visual design and process structuring capabilities which allow users to communicate with the machine on a conceptual level, rather than being forced to address details of underlying implementation technology.

1.1.5 Knowledge Transfer Systems

Decision support in the context of this study includes the transfer of design and process rules from developers to users, via computer-mediated tools. The tools in effect sponsor developers' process models as appropriate and efficient means to develop high quality software. In this sense, learning which takes place during tool use constitutes knowledge transfer, not merely process assistance.

1.2 Other Work In This Series

Related work by this author includes a doctoral dissertation proposal [1], a paper in the *Electronic Journal of Communication* [2], a presentation at the Symposium on Human Factors in Information Systems [3], and a special issue of the *Electronic Journal on Virtual Culture* [4].

2 Domain

2.1 Domain Generalization

Although the domain featured in this study is software engineering, the conceptual framework, analytic approach, and understanding of user responses developed here may facilitate attempts to refine computer-mediated design tools used in other domains (e.g., publication design, mechanical engineering, architecture, industrial chemistry and bioengineering).

2.2 Domain Selection and Characteristics

Software engineering is an appropriate first domain for the study of computer-mediated design tools. Computer-aided software engineering (CASE) tools are used to create software packages, including other tools. The creation of tools for other

domains allows CASE tools to influence design activities in a number of professions and industries.

The software engineering domain is populated by a wide variety of creative, perceptive, process-oriented and logical individuals who are especially capable critics of the tools which they use. Since they themselves are software developers, users in this domain have the experience and knowledge to evaluate CASE tools in detail. In significant ways, design tool users in other domains are somewhat like typical motorists, who may know how their cars should perform, but would not be capable of criticizing internal engineering features of automobile engines.

Design activities in the study domain take place amidst a host of sometimes conflicting priorities, sponsored by a number of major actors. Stakeholders include customers for software created with CASE tools, tool developers, tool users, user management, and developer management. This study focuses on tool developers and tool users. Developers' perceptions are influenced by concerns about risk assessment, resource management, process control, quality assurance, style preferences, and tool user satisfaction. Users' perceptions are affected by project requirements, training and design experience, the work environment, style preferences, cost and time-on-task (schedule), and satisfaction with the tools.

2.3 Computer-Aided Software Engineering Tools

CASE tools facilitate the creation and implementation of design elements and the generation of executable software. They do so in part by implementing constraints made necessary by a variety of economic, quality, schedule and maintainability concerns. The tools make it possible for system analysts, software engineers and programmers to access a common, current, and comprehensive repository of design and implementation information, thereby facilitating cooperation essential to the creation of quality software on large, complex projects.

CASE tools range from single-purpose "programmer's workbench" modules to intertwined hierarchical or object-oriented toolsets which facilitate process and quality control throughout much of the software development life cycle. Included are tools which support planning, analysis and design; user interface development; code generation; reverse engineering; maintenance; test and evaluation; configuration management, and documentation.

3 Conceptual Framework

3.1 Mental Models, Creativity and Standards

Computerized design tools provide a meeting ground between users' and developers' mental models of desirable processes, functions, tasks and constraints.

It is a premise of this study that user responses to design tools may be affected substantially by the tools' constraint management systems and by the degree of match (cognitive fit) between the mental models of tool developers and tool users.

Conflicts sometimes occur between users' sense of design creativity and the need to protect product and process standards via constraints. These conflicts are due in part to a mismatch between values and procedures preferred by tool developers versus tool users. This study proposes that this mismatch is in fact misunderstanding caused by incomplete cognitive fit between the mental models of developers and tool users. It seeks to improve our understanding of design decision

support, the communication of mental models, and user behavior, so as to facilitate design creativity while ensuring reasonable adherence to product and process standards.

3.2 The Transfer of Knowledge

The transfer of knowledge during CASE tool use (Section 1.1.5) is hampered not only by creativity and constraint conflicts (Section 3.1), but also by inherent limitations of cognitive transfer processes. For example, researchers [5] have found that under certain circumstances some subjects have difficulty generalizing learning. The unique context of many design decisions (i.e., situational specificity) frustrates tool users' attempts to generalize process models from prior experience. Realizing this, some tool developers may favor inflexible constraints because they feel it is better to limit user creativity than to risk degradation of end-product quality because of inappropriate user generalization.

3.3 Constraint Issues and Typology

The constraint management system (CMS) within a CASE tool represents the weighted process and product quality requirements associated with each stage in the software development life cycle. A properly implemented CMS can be a valuable decision support feature, guiding users in the selection of design options which are consistent with requirements.

In many instances, a user may be involved in only a few stages of product development. Since the CMS represents requirements of all stages, it can help improve product quality and reduce error by evaluating user actions in the context of the project as a whole, compensating for the inherently narrow view of a user involved in only a few stages. From the user viewpoint, however, constraints can be frustrating precisely because they implement concerns from outside of the immediate task environment.

In order to study constraint management systematically and rigorously, a typology of constraints is necessary. A carefully constructed typology reduces ambiguity, facilitates creation of appropriate measurement instruments, and helps to validate the analytic framework. The typology devised for this study consists of four discontinuous axes. It seeks to characterize the key attributes of any constraint implemented by a developer in a tool. The first axis is a human-computer control scale (Table 3.3); the second consists of implementation styles; the third includes the sources of authority for requirements which trigger constraints; the fourth is the relative importance (weight) of the constraint (i.e., the severity of impact if the constraint is violated).

Constraint satisfaction does not necessarily require the application of inflexible rules; it only mandates that a series of decisions taken as a whole meets necessary and sufficient standards. It is end-product characteristics, not the constraints intended to ensure those characteristics, that are important. Therefore, constraint flexibility does not necessarily result in product degradation. Without an appreciation of this aspect of process control, constraints implemented by developers in design tools can be unnecessarily restrictive.

An analogy from elsewhere within software engineering might be that of critical path method (CPM) scheduling. Under CPM, project milestones are plotted sequentially in a chronological graph. Some of the activities represented are

Table 3.3. A Human-Computer Control Scale

1.	Human considers alternatives, makes and implements decision.
2.	Computer offers a set of alternatives which human may ignore in making decision.
3.	Computer offers a restricted set of alternatives, and human decides which to implement.
4.	Computer offers a restricted set of alternatives and suggests one, but human still makes and implements final decision.
5.	Computer offers a restricted set of alternatives and suggests one which it will implement if human approves.
6.	Computer makes decision but gives human option to veto before implementation.
7.	Computer makes and implements decision, but must inform human after the fact.
8.	Computer makes and implements decision, and informs human only if asked to.
9.	Computer makes and implements decision, and informs human only if it feels this is warranted.
10.	Computer makes and implements decision if it feels it should, and informs human only if it feels this is warranted.

[a]Adapted from [6].

performed concurrently by several individuals on the project. Some tasks take longer than others; certain (antecedent) tasks must be completed before (dependent) others can be started. If it were to take longer than scheduled to complete an antecedent task, the delay would ripple through following dependent tasks and place final delivery off schedule. Antecedent-dependent tasks are on the "critical path". Other tasks not on the critical path can take longer than anticipated without serious repercussion, just as constraints which are not intrinsically vital to product quality, cost or schedule can be relaxed.

3.4 The Problem Space

The problem space within which this study is being conducted may be envisioned as two interrelated conceptual graphs characterizing several key aspects of computer-mediated design activities.

The first graph, "Process Engineering", describes design functions and tasks on one axis and constraints on the other. Design activities are described as functions comprised of component tasks. Constraints are characterized by a four-axis typology (Section 3.3), and are arrayed against the constraints to which they might apply (Figure 3.4(a)).

The second graph, "Human Factors", is a graph which describes groups of independent variables which might impact user behavior on one axis, and user behavioral and perceptual responses (dependent variables) on the other (Figure 3.4(b)).

4 Relevant Literature

This study lies at the intersection of several disciplines in the humanities and applied sciences. Included are information science, psychology, computer science, engineering and management science. Information science addresses man-machine

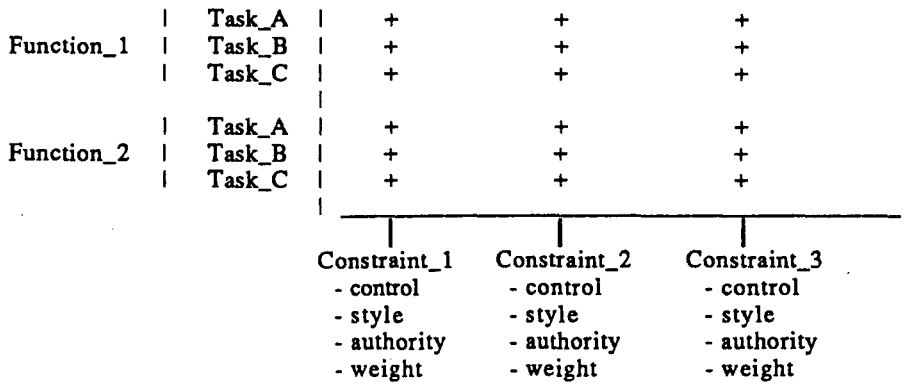


Fig. 3.4(a). The Process Engineering Problem Space

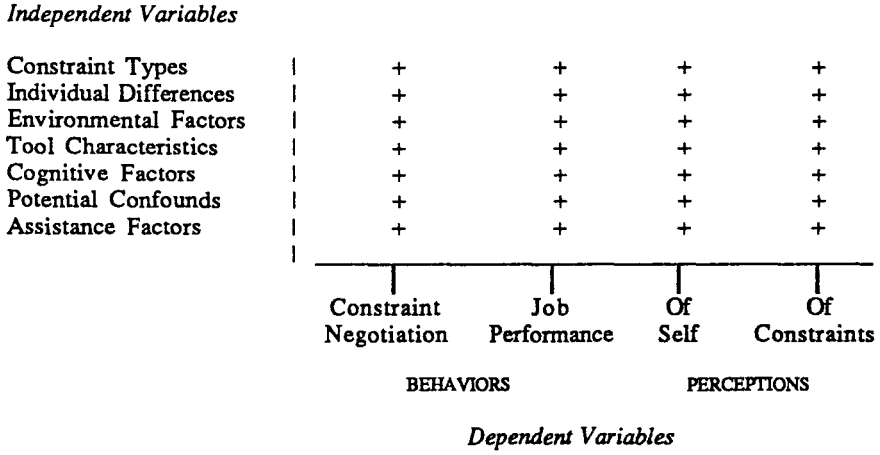


Fig. 3.4(b). The Human Factors Problem Space

systems (e.g., human-computer interaction); psychology contributes cognition (e.g., mental modeling); computer science offers software engineering; management science provides decision support, and engineering adds process design and control (e.g., constraint management).

4.1 Sources and Authors

Conference proceedings and journals contain the majority of material relevant to this study. Especially relevant are the annual proceedings of the ACM SIGCHI conferences, the triennial INTERACT meetings (Europe), and the biennial symposia on Human Factors in Information Systems (U.S.). Key journals include *Behaviour & Information Technology*, the *International Journal of Man-Machine Studies*, *Software Engineering Notes*, the *Software Engineering Journal*, *ACM Computing Surveys*, *IEEE Software*, *Management Science*, *Decision Sciences*, and *Communications of the ACM*. The key publisher was Elsevier/North-Holland.

Of the scores of authors whose work has contributed thus far to this study, a few whose research seemed especially pertinent are listed in Table 4.1, "Key Authors".

Table 4.1. Key Authors

Author	Area of Contribution
Ackermann, D.	mental models
Bullinger., H.-J.	interface techniques
Carroll, J.	intelligent advising
Curtis, B.	process modeling
Dechter, R.	constraint networks
Diaper, D.	task analysis
Fishbein, M.	user satisfaction
Galetta, D.	cognitive fit
Martin, J.	information engineering
Norman, D.	user error behavior
Rasmussen, J.	human information processing
Rouse, W.	adaptive advising
Salvendy, G.	performance modeling
Silver, M.	user decision behavior
Wasserman, A.	automated tools
Woods, D.	decision aids

4.2 Representative Works

The literature relevant to this study has been organized into two units dealing with (a) the design activity context and (b) user responses to constraint management (Figure 4.2, "Organization of the Literature"). Space limitations prevent detailed discussion of the literature. However, representative works are cited in "References" (Process Modeling and Design [7-17], Domain [18-38], Technologies [39-95], Cognitive Factors [96-120], Potential Confounds [121-135], Behaviors [136-145], and Perceptions [146-154]).

5 Research Questions and Methodology

5.1 Research Questions

1. How are characteristics of constraint management within CASE tools (e.g., flexibility) associated with user behavior and perceptions in response to such tools?

2. To what extent are factors such as cognitive fit, trust, mental workload, complexity and users' and developers' mental models associated with user behavior and perceptions in response to CASE tools?

3. How might potential workplace confounds contaminate apparent relationships among constraint management, cognitive factors, and user behavior and perceptions?

4. To what degree does interaction with a design decision support system educate users regarding principles of software development, and equip them to negotiate constraints successfully?

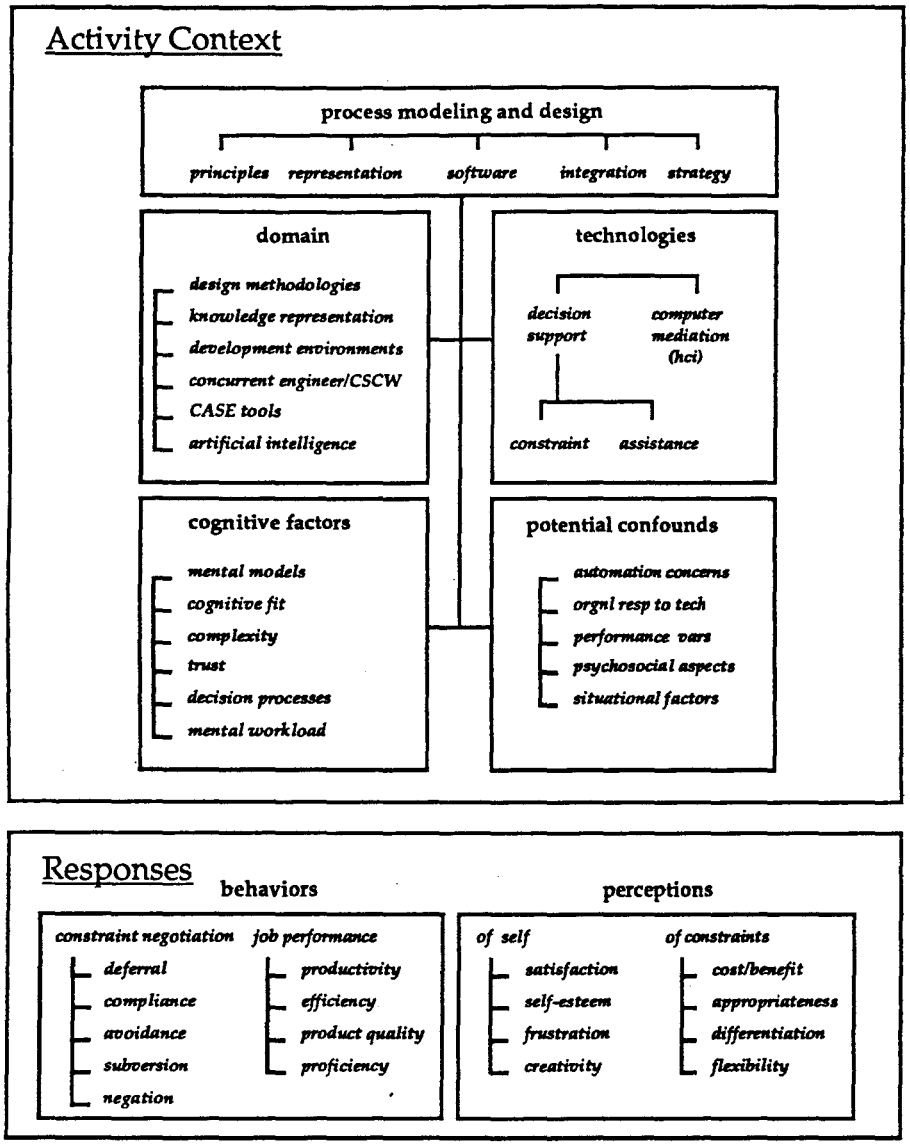


Fig. 4.2. Organization of the Literature

5.2 Methodology

5.2.1 Experimental Design

The first of three stages planned for the study is currently underway: Interviews of key stakeholders in the computer-mediated design process. In Stage 2, tool users and non-tool software developers (a control group) will be asked to complete a survey questionnaire about their behavior and perceptions; in Stage 3, several tool users will be observed as they execute a standard script of design tasks.

5.2.2 Subjects

This study examines computer-mediated cooperation among skilled users who create program modules as end-products, software engineers and system analysts who define intermodule relationships, set schedules and monitor quality assurance, and developers who create the CASE tools which the first two groups use.

Users apply CASE tools in the design, fabrication, testing, integration, maintenance and enhancement of computer programs. Each user has his or her own professional perspective regarding appropriate models and procedures for software development. Some feel that their trade is an art more than a science, and resist the application of constraints which restrict their options.

Tool developers generally are highly skilled software engineers and system analysts employed by firms whose marketable products are CASE tools. They often are not the ultimate authority for requirements which they implement as constraints. However, the details of constraint implementation are their responsibility.

Developers define the processes, functions, tasks and constraints incorporated in a CASE toolset; users select and manipulate functions and tasks in a process sequence to create a product (executable software).

5.2.3 Data Collection

A structured interview guideline of 12 questions was drafted, based upon a conceptual analysis of the problem area and upon a fully defined and operationalized list of variables. Data are being collected in private interviews with user managers, team leaders and analysts at two Fortune 500 firms, and tool developers at the software engineering unit of a national information systems company. Interviews are being audio recorded and transcribed, for verification and analysis.

5.2.4 Analysis

Transcripts are being separated into sections corresponding to the interview outline structure. Next, responses to similar questions will be compared and aggregated. Common and contrasting themes will be identified and matched to the research questions. (Preliminary findings related to the third research question will be reported in [3].) Where possible, preliminary conclusions will be drawn. Refinements needed in the instruments will be noted, and themes for future research recorded.

6 Conclusion

Automated design has become common in several professions. This study of constraints, user behavior and the transfer of mental models incorporates many concerns characteristic of any process control or decision support situation. Improved understanding of constraint management problems, user behavior and perceptual responses, and the coupling of requirements to constraint implementation can be of considerable value to any enterprise.

References[3]

1. Day, D.: Behavioral and Perceptual Responses to Constraint Management in Computer-Mediated Design Activities: A Proposal For Research. Unpublished dissertation proposal, School of Information Studies, Syracuse University 1993
2. Day, D.: Precis of behavioral and perceptual responses to constraint management in computer-mediated design activities. *Electronic Journal of Communication* 3:2 (1993)
3. Day, D.: Contextual aspects of CASE tool use in information system design and development. (To be presented at the Fifth Symposium on Human Factors in Information Systems, Cleveland (October 1993))
4. Day, D. (ed.): Computerized Tools As Intermediaries in the Communication of Mental Maps. Special issue of the *Electronic Journal on Virtual Culture* (in preparation)
5. Singley, M., Anderson, J.: *The Transfer of Cognitive Skill.* Cambridge, Mass.: Harvard University Press 1989
6. Sheridan, T.: Computer control and human alienation. *Technology Review* 83:1 (October), 65-67, 71-73 (1980)

Process Modeling and Design

7. Bras, B., Mistree, F.: Designing design processes in decision-based concurrent engineering. In: *Aerospace Product/Process Design Interface,* 15-36. Warrendale, Pa.: Society of Automotive Engineers 1991
8. Curtis, B., Kellner, M., Over, J.: Process modeling. *CACM* 35, 75-90 (1992)
9. Gibson, D., Salvendy, G.: Knowledge representation in human problem solving. *BIT* 9, 191-200 (1990)
10. Newell, A., Simon, H.: *Human Problem Solving.* Englewood Cliffs, N.J.: Prentice-Hall 1972
11. Norman, D.: Design rules based on analyses of human error. *CACM* 26, 254-258 (1983)
12. Peckham, J., Maryanski, F.: Semantic data models. *ACS* 20, 153-189 (1988)
13. Sharp, H.: The role of domain knowledge in software design. *BIT* 10, 383-401 (1991)
14. Smith, F.: Towards a heuristic theory of problem structuring. *Management Science* 34, 1489-1506 (1988)

---

[3] Abbreviations are as follows. ACS: *ACM Computing Surveys*; ACM: Association for Computing Machinery; ASME: American Society of Mechanical Engineers; BIT: *Behaviour & Information Technology*; CACM: *Communications of the ACM*; CUP: Cambridge University Press; DSI: Decision Sciences Institute; ERIC: Educational Resources Information Center; DSS: *Decision Support Systems*; IEEE: Institute of Electrical and Electronics Engineers; IJMMS: *International Journal of Man-Machine Studies*; MIT: Massachusetts Institute of Technology; SEN: *Software Engineering Notes*; SEJ: *Software Engineering Journal*

15. Sonnenwald, D.: Developing a theory to guide the process of designing. Proceedings, 15th International SIGIR'92, 310-317, Copenhagen. New York: ACM 1992
16. Todd, P.: Process tracing methods in DSS research. *MIS Quarterly* 11, 493-512 (1987)
17. Webster, D.: Mapping the design information representation terrain. *IEEE Computer* 21, 8-23 (1988)

Domain

18. Baroudi, J., Ginzberg, M.: Impact of the technological environment on programmer/analyst job outcomes. *CACM* 29, 546-555 (1986)
19. Baxter, I.: Design maintenance systems. *CACM* 35, 73-89 (1992)
20. Davies, S.: Characterizing the program design activity. *BIT* 10, 173-190 (1991)
21. Deepak, J., Maher, M.: Combining expert systems and CAD techniques. In: *Artificial Intelligence Developments and Applications*, 65-81. Amsterdam: North-Holland 1988
22. Fisher, A.: *CASE: Using Software Development Tools* (2d. ed.). New York: Wiley 1991
23. Forte, G.: Tools fair: Out of the lab, onto the shelf. *IEEE Software* 9, 70-79 (1992)
24. Houghton, R.: Annotated bibliography on software engineering environments. *SEN 10*, 62-76 (1985)
25. Humphrey, W.: Software and the factory paradigm. *SEJ* 6, 370-376 (1991)
26. Jones, M., Arnett, K.: CASE use is growing, but in surprising ways. *Datamation* 38:10, 108-109 (1992)
27. Kemerer, C.: How the learning curve affects CASE tool adoption. *IEEE Software* 9, 23-28 (1992)
28. Klaczko-Ryndziun, S., Goller, M.: Automatic reasoning for creativeness decision support. In.: *Human Aspects in Computing*, 894-898. Amsterdam: Elsevier 1991
29. Liker, J., Fleisher, M., et al: Designers and their machines: CAD use and support in the U.S. *CACM* 35, 77-95 (1992)
30. MacDonald, I.: Automating Information Engineering. In: D. Benyon, S. Skidmore (eds.) *Automating Systems Development*. New York: Plenum Press 1988
31. Majchrzak, A., Collins, P., Mandeville, D.: A quantitative assessment of changes in work activities resulting from computer-aided design. *BIT* 5, 259-271 (1986)
32. Martin, J.: *Computer-Aided Software Engineering*. Marblehead, Mass.: James Martin Report 1987
33. Minsky, M.: A framework for representing knowledge. In: *Mind Design: Philosophy, Psychology and Artificial Intelligence*. Montgomery, Ala.: Bradford Books 1980
34. Perlman, G.: Software tools for user interface development. In.: M. Helander (ed.) *The Handbook of Human Computer Interaction*. Amsterdam: North-Holland 1988
35. Seppanen, V., Heikkinen, M., Lintulampi, R.: SPADE - Towards CASE tools that can guide design. In: R. Andersen, J. Bubenko, A. Solvberg, *Advanced Information Systems*, 222-239. Berlin: Springer-Verlag 1991
36. Stefik, M.: Planning with constraints. *Artificial Intelligence* 16, 111-140 (1981)
37. The, L.: Bridging the CASE/OOP gap. *Datamation* 38:5, 63-64 (1992)
38. Wasserman, A., Pircher, P., et al: Developing interactive systems with User Software Engineering methodology. *IEEE Transactions on Software Engineering* 12, 326-345 (1986)

39. Ambardar, A.: Human-computer interaction and individual differences. In: G. Salvendy (ed.) *Human-Computer Interaction*, 207-211. Amsterdam: Elsevier 1984
40. Baeker, R., Buxton, W. (eds.): *Readings in Human-Computer Interaction.* Los Altos, Calif.: Morgan Kaufman 1987
41. Billman, B., Courtney, J.: Automated discovery in managerial problem solving: Formation of causal hypotheses for cognitive mapping. *Decision Sciences* 24, 23-41 (1993)
42. Boettcher, K., Tenney, R.: Distributed decisionmaking with constrained decisionmakers. *IEEE Transactions on Systems, Man & Cybernetics* 16, 813-823 (1986)
43. Borning, A., Duisberg, R., et al: Constraint hierarchies. *SIGPLAN Notices* 22, 48-60 (1987)
44. Bowne, J., O'Grady, P.: A technology for building life-cycle design advisers. In: G. Kinzel, S. Rohde (eds.) *Computers in Engineering 1990*, 1-7. New York: ASME 1990
45. Browne, D., Norman, M.: *Adaptive User Interfaces.* London: Academic Press 1992
46. Carroll, J., Aaronson, A.: Learning by doing with simulated intelligent help. *CACM* 31, 1064-1079 (1988)
47. Carroll, J., McKendree, J.: Interface design issues for advice-giving expert systems. *CACM* 38, 14-31 (1987)
48. Cesta, A., Romano, G.: Explanations in an intelligent help system. In: H.-J. Bullinger (ed.) *Human Aspects in Computing.* Amsterdam: Elsevier 1991
49. Curtis, B.: *Human Factors in Software Development.* Washington, D.C.: IEEE Computer Society 1985
50. Darses, F.: The constraint satisfaction approach to design: A psychological investigation. *Acta Psychologica* 78, 307-325 (1991)
51. Dechter, R. (ed.): *Working Notes of the 1991 Spring Symposium on Constraint-Based Reasoning.* Tech. Rpt. 91-65, University. of California. at Irvine. American Association for Artificial Intelligence 1991
52. Duncan, N., Paradice, D.: Identifying creativity enhancing features for expert systems. Proceedings, Decision Sciences Institute, 591-593 Atlanta: DSI 1992
53. Elam, J.: Can software influence creativity. *Information Systems Research* 1, 1-22 (1990)
54. Ettema, J.: Studies in creativity and constraint. ERIC ED 259 367 (1985)
55. Fischer, G., Mastaglio, T.: A conceptual framework for knowledge-based critic systems. *DSS* 7, 355-378 (1991)
56. Freedy, A., Madni, A., Samet, M.: Adaptive user models: Methodology and applications for man-machine systems. In: Rouse, W. (ed.) *Advances in Man-Machine Systems Research* 2, 249-293. Greenwich, Conn.: JAI Press 1985
57. Gould, J., Lewis, C.: Designing for usability: Key principles and what designers think. *CACM* 28, 300-311 (1985)
58. Guindon, R., Curtis, B.: Control of cognitive processes during software design: What tools are needed? In: *Human Factors in Computing Systems CHI'88*, 263-268. Proceedings 1988. New York: ACM
59. Hancock, P., Chignell, M.: *Intelligent Interfaces.* Amsterdam: North-Holland 1989
60. Harrison, M., Thimbleby, H.: *Formal Methods in Human-Computer Interaction.* Cambridge, UK: CUP 1990
61. Hartson, H.: Human-computer interface development: Concepts and systems. *ACS* 21, 5-92 (1989)
62. Holtzman, S.: *Intelligent Decision Systems.* Reading, Mass.: Addison-Wesley 1989

63. Jacob, V., Pakath, R.: The roles of computerized support systems: A decision subprocess-based analysis. *BIT* 10, 231-252 (1991)
64. Kaiser, G., Feiler, P., Popovich, S.: Intelligent assistance for software development and maintenance. *IEEE Software* 5, 40-49 (1988)
65. Kieras, D., Polson, P.: An approach to the formal analysis of user complexity. *IJMMS* 22, 365-394 (1985)
66. Kling, R., Iacono, S.: The control of information systems developments after implementation. *CACM* 27, 1218-1226 (1984)
67. Laurel, B.: Constraints. In: B. Laurel, *Computers As Theatre*, 99-112. Reading, Mass.: Addison-Wesley 1991
68. Liang, T.-P., Jones, C.: Design of a self-evolving decision support system. *Journal of Management Information Systems* 4, 59-82 (1987)
69. Mackay, J., Barr, S., Kletke, M.: An empirical investigation of the effects of decision aids on problem-solving processes. *Decision Sciences* 23, 648-672 (1992)
70. Mackworth, A.: Constraint satisfaction. In: S. Shapiro (ed.) *The Encyclopedia of Artificial Intelligence*, 205-211. New York: Wiley 1987
71. Marxhall, R., Bregant, G.: The Overseer: An approach to design management. In: F. Rammig (ed.) *Tool Integration and Design Environments*, 289-309. Amsterdam: Elsevier 1988
72. McKendree, J., Zaback, J.: Planning for advising. In: E. Soloway, D. Frye, S. Sheppard (eds.) *Human Factors in Computing Systems CHI'88*, 179-183. New York: ACM 1988
73. Moffitt, K., Hershauer, J., Reneau, H.: Knowledge transfer from interacting with expert system explanation facilities. In: J. Carey (ed.) *Human Factors in Information Systems: An Organizational Perspective*, 253-269. Norwood, N.J.: Ablex 1991
74. Nielsen, J.: *Usability Engineering*. London: Academic Press 1992
75. Papstein, P., Frese, M.: Transferring skills from training to the actual work situation: The role of task application knowledge, action styles and job decision lattitude. In: Soloway, E., Frye, D. and Sheppard, S. (eds.) *Human Factors in Computing Systems CHI'88*, 55-60. Proceedings, Washington, D.C. 1988. New York: ACM
76. Potosnak, K.: Mental models: Helping users understand software. *IEEE Software* 6, 85-88 (1989)
77. Preece, J., Keller, L.: *Human-Computer Interaction, Selected Readings*. Englewood Cliffs, N.J.: Prentice-Hall 1989
78. Rasmussen, J.: *Information Processing and Human-Machine Interaction*. New York: North-Holland 1986
79. Rouse, W.: Adaptive aiding for human/computer control. *Human Factors* 30, 431-443 (1988)
80. Sata, T., Warman, E. (ed.): *Man-Machine Communication in CAD/CAM*. Proceedings, IFIP WG 5.2-5.3, Tokyo 1980. Amsterdam: North-Holland 1981
81. Savolainen, T.: Expanding human-computer interaction by computer-aided creativity. *Interacting with Computers* 2, 161-174 (1990)
82. Schauer, H.: What kind of involvement do we envisage for designers and users. In: Briefs, U., Tagg, E. (eds.) *Education for System Designer/User Cooperation*, 13-17. Amsterdam: Elsevier 1985
83. Serrano, D.: Managing constraints in concurrent design: First steps. *Computers in Engineering 1990*, 159-164. New York: ASME 1990
84. Shackel, B. (ed.): *Human-Computer Interaction: Interact '84*. Amsterdam: North-Holland 1985
85. Shneiderman, B.: *Designing the User Interface*. Reading, Mass.: Addison-Wesley 1987
86. Silver, M.: *Systems That Support Decision Makers*. Chichester, UK: Wiley 1991

87. Silverman, B.: Survey of expert critiquing systems. *CACM* 35, 106-127 (1992)
88. Stohr, E., Konsynski, B.: *Information Systems and Decision Processes.* New York: IEEE
89. Tan, B., Lo, T.: The impact of interface customization on the effect of cognitive style on information system success. *BIT* 10, 297-310 (1991)
90. Taylor, R.: Perception of problem constraints. *Management Science* 22, 22-29 (1975)
91. Todd, P., Benbasat, I.: An experimental investigation of the impact of computer based decision aids on decision making strategies. *Information Systems Research* 2, 87-115 (1991)
92. Weber, D.: Information management: Combining computer-controlled with human-controlled functions in CIM. In: H.-J. Bullinger (ed.) *Human Aspects in Computing*, 1026-1027. Amsterdam: Elsevier 1991
93. Wetzenstein-Ollenschlager, E., Wandke, H.: Toward adaptive human-computer interfaces. In: D. Ackermann, M. Tauber (eds.) *Mental Models and Human-Computer Interaction 1*, 231-252. Amsterdam: Elsevier 1990
94. Williges, R., Elkerton, J., et al: User assistance in human-computer interfaces. In: R. Ehrich, R. Williges (eds.) *Human-Computer Dialogue Design*, 291-317. Amsterdam: Elsevier 1986
95. Zmud, R.: Individual differences and MIS success: A review of empirical literature. *Management Science* 25, 966-979 (1979)

Cognitive Factors

96. Anon.: Mental workload: Research on CAD work and on the implementation of office automation. Espoo, Finland: Helsinki University of Technology. NTIS PB92127372XSP, May 24, 1991
97. Ackermann, D., Tauber, M.: *Mental Models and Human-Computer Interaction.* Amsterdam: North-Holland 1990
98. Barki, H., Huff, S.: Change, attitude to change, and decision support system success. *Information and Management* 9, 261-268 (1985)
99. Bloomfield, B.: Understanding the social practices of systems developers. *Journal of Information Systems* 2, 189-206 (1992)
100. Brookes, R.: Towards a theory of the cognitive processes in computer programming. *IJMMS* 9, 737-751 (1977)
101. Carey, J.: The issue of cognitive style in MIS/DSS Research. In: J. Carey (ed.) *Human Factors in Information Systems: An Organizational Perspective*, 337-348. Norwood, N.J.: Ablex 1991
102. Carroll, J.: *Interfacing Thought.* Cambridge, Mass.: MIT Press 1987
103. Dagwell, R., Weber, R.: Systems designer's user models: A comparative study and methodology. *CACM* 26, 987-997 (1983)
104. Darses, F.: Constraints in design: Towards a method of psychological analysis. In: Diaper, D., Gilmore, D., et al *Human-Computer Interaction.*, 135-139. Proceedings, INTERACT'90. Amsterdam: North-Holland 1990
105. Gardiner, M., Christie, B.: *Applying Cognitive Psychology to User-Interface Design.* Chichester, UK: Wiley 1987
106. Hirschheim, R.: The effect of a priori views on the social implications of computing: The case of office automation. *ACS* 18, 166-195 (1986)
107. Hockey, R.: Styles, skills and strategies: Cognitive variability and its implications for the role of mental models in HCI. In: D. Ackermann, M. Tauber (eds.) *Mental Models and Human-Computer Interaction 1*, 113-129. Amsterdam: Elsevier 1990
108. Kydd, C.: Cognitive biases in the use of computer-based DSS. *Omega* 17, 335-344 (1989)
109. Lindsay, P., Norman, D.: *Human Information Processing.* New York: Academic Press 1977

110. Muir, B.: Trust between humans and machines, and the design of decision aids. *IJMMS* 27, 527-539 (1987)
111. Norman, D. Some observations on mental models. In: A. Stevens, D. Gentner (eds.) *Mental Models*, 7-14. Hillsdale, N.J.: Lawrence Erlbaum 1983
112. Orlikowski, W.: Social implications of CASE tools for systems developers. Proceedings, Boston 1989, 199-210. 10th International Conference on Information Systems
113. Rasmussen, J.: A cognitive engineering approach to the modeling of decision making. In: Rouse, W. (ed.) *Advances in Man-Machine Systems Research* 4, 165-243. Greenwich, Conn.: JAI Press 1988
114. Rosson, M., Maass, S.: The designer as user: Building requirements for design tools from design practice. *CACM* 31, 1288-1298 (1988)
115. Suchman, L.: *Plans and Situated Actions: The Problem of Human-Machine Communication*. Cambridge, UK: CUP 1987
116. Vessey, I., Galletta, D.: Cognitive fit: An empirical study of information acquisition. *Information Systems Research* 2, 63-84 (1991)
117. Waern, Y.: *Cognitive Aspects of Computer Supported Tasks*. Chichester, UK: Wiley 1989
118. Winograd, T., Flores, F.: *Understanding Computers and Cognition*. Wokingham, UK: Addison-Wesley 1986
119. Woods, D.: Cognitive engineering: Human problem solving with tools. *Human Factors* 30, 415-430 (1988)
120. Woods, D.: Coping with complexity. In: Goodstein, L., Andersen, H., and Olsen, S. (eds.) *Mental Models, Tasks and Errors*. London: Taylor & Francis 1988

Potential Confounds

121. Day, D.: Human factors issues in the electronic newsroom. In: M. Nurminen, G. Weir (eds.) *Human Jobs and Computer Interfaces*, 81-100. Proceedings, Tampere, Finland 1991. IFIP WG 9.1 Working Conference. Amsterdam: North-Holland 1991
122. Springer, J., Langner, T., et al: Experimental comparison of CAD systems by stressor variables. *International Journal of Human-Computer Interaction* 3, 375-405 (1991)
123. Taylor, R.: Information use environments. *Progress in Communication Science* 10, 217-255 (1991)
124. Cohen, B.: Psychosocial environments created by computer use for managers and systems analysts. *Human-Computer Interaction*, 379-384. Amsterdam: Elsevier 1984
125. Dawson, P., McLaughlin, I.: Computer technology and the redefinition of supervision. *Journal of Management Studies* 23, 116-132 (1986)
126. Karake, Z.: *Information Technology and Management Control: An Agency Theory Perspective*. New York: Praeger 1992
127. Kling, R.:, Scacchi, W. The web of computing: Computer technology as social organization. *Advances in Computers* 21, 1-90 (1982)
128. Leifer, R., McDonough, E.: Computerization as a predominant technology affecting work group structure. Proceedings, International Conference on Information Systems 1980, 238-248
129. Markus, L.: Power, politics and MIS implementation. *CACM* 26, 430-444 (1983)
130. Srinivasan, A., Kaiser, K.: Relationships between selected organizational factors and systems development. *CACM* 30, 556-562 (1987)
131. Turner, J.: Computer mediated work: The interplay between technology and structured jobs. *CACM* 27, 1210-12 (1984)
132. Weber, R.: Computer technology and jobs: An impact assessment model. *CACM* 31, 68-77 (1988)

133. Wisner, A., Daniellou, F., et al: Place of work analysis in software design. In: G. Salvendy (ed.) *Human-Computer Interaction*, 147-156. Amsterdam: Elsevier 1984
134. Yaverbaum, G.: Critical factors in the user environment: An experimental study of users, organizations, and task. *MIS Quarterly* 12, 75-88 (1988)
135. Zuboff, S.: *In the Age of the Smart Machine: The Future of Work*. New York: Basic Books 1988

Behaviors

136. Bailey, R.: *Human Error in Computer Systems*. Englewood Cliffs, N.J.: Prentice-Hall 1983
137. Banker, R., Datar, S., Kemerer, C.: A model to evaluate variables impacting productivity on software maintenance. *Management Science* 37, 1-18 (1991)
138. Ciampi, A., Silberfield, M.: Measurement of individual preferences: The importance of situation-specific variables. *Medical Decision-Making* 2, 483-495 (1982)
139. Czaja, S., Hammond, K., et al.: Aged related differences in learning to use a text-editing system. *BIT* 8, 309-319 (1989)
140. Egan, D.: Individual differences in human computer interaction. In: M. Helander (ed.) *Handbook of Human Computer Interaction*, 543-568. Amsterdam: Elsevier 1988
141. Einhorn, H., Hogarth, R.: Behavioral decision theory: Process of judgment and choice. *Annual Review of Psychology* 32, 52-88 (1981)
142. Payne, J.: Contingent decision behavior. *Psychological Bulletin* 92, 382-402 (1982)
143. Rasch, R., Tosi, H.: Factors affecting software developers' performance. *MIS Quarterly* 16, 395-409 (1992)
144. Wickens, C.: *Engineering Psychology and Human Performance*. Columbus, Ohio: Merrill 1984
145. Zavala, A.: Stress and factors of productivity among software development workers. In: G. Salvendy (ed.) *Human-Computer Interaction*, 365-370. Amsterdam: Elsevier 1984

Perceptions

146. Bittner, U., Schnath, J., Hesse, W.: Werkzeugeinsatz bei der entwicklung von software-systemen. *Software Technik Trends* 13, 14-23 (1993)
147. Fishbein, M.: Attitude and the prediction of behavior. In: Fishbein, M. (ed.) *Readings in Attitude Theory and Measurement*. New York: Wiley 1967
148. Hanson, S., Rosinski, R.: Programmer perceptions of productivity and programming tools. *CACM* 28, 180-189 (1985)
149. Huuhtanen, P., Seitsamo, J.: Changes in mastery of computer applications and endusers' evaluations of the implementation process and psychological well-being. In: H.J. Bullinger (ed.) *Human Aspects in Computing: Design and Use of Interactive Systems and Information Management*, 1075-1079. Amsterdam: Elsevier 1991
150. Knauth, P., Joseph, J., Gemunden, H.: User acceptance of computer aided design (CAD). In: H.-J. Bullinger (ed.) *Human Aspects in Computing: Design and Use of Interactive Systems and Information Management*, 1070-1074. Amsterdam: Elsevier 1991
151. Melone, N.: A theoretical assessment of the user-satisfaction construct in information systems research. *Management Science* 36, 76-91 (1990)
152. Norman, R., Nunamaker, J.: CASE productivity perceptions of software engineering professionals. *CACM* 32, 1102-1108 (1989)
153. Rushinek, A., Rushinek, S.: What makes users happy? *CACM* 29, 594-598 (1986)
154. Salancik, G., Pfeffer, J.: An examination of need-satisfaction models of job attitudes. *Administrative Science Quarterly* 22, 427-456 (1977)

A design for accessing multimedia information using cohesion

Roger Espinosa and Patricia Baggett

School of Education, University of Michigan, Ann Arbor, MI 48109-1259, USA

Keywords. Accessing information, hypermedia, multimedia, cohesion

Abstract. This paper presents a design for multimedia information access based on cohesion and reports on some implementations giving it a limited test. Elements in a database are cohesive if they refer to the same concept. The designer selects cohesive elements and sequentially links those referring to the same concept, forming a "subway line" for each concept. Cohesive elements can be in different modalities, e.g., graphics, text, photos. Users are given access via the "subway lines." At a given "stop" the user can either follow the subway line he or she is currently on (by default), or change to another subway line which crosses that stop but which links elements referring to a different concept. At each stop, a list of concepts, and thus subway lines crossing at the stop, can be made available for the user, acting as an index of currently active concepts. This method of accessing data was designed as an alternative to menus and keyword approaches. We have implemented the method in several domains, including a repair task, literature analysis, a history of mathematics, and elementary mathematics lesson plans.

1. Overview

1.1. The problem

Often when one is presented with a large amount of computerized information, one is faced with developing a way for it to be browsed. This paper describes a design for accessing computerized information. The information to be browsed can be lengthy and can consist of material from different modalities, such as text and pictures.

A unique feature of the approach given here is that after the material is divided into units by the designer, access links between units are based on cohesive elements in the material. Two elements are cohesive if they refer to the same concept, even if they occur in different modalities. Users can follow "subway lines" through the material; each line sequentially links elements referring to the same concept. At a given "stop" the user can either follow the subway line he or she is currently on (by default), or change to another subway line which crosses that stop but which links elements referring to a different concept.

1.2. Why this approach?

1.2.1. It is different from a hierarchical or menu structure and is advantageous for loosely connected heterogeneous material. It is also convenient for hierarchical material in which the hierarchy is not known to the designer.

1.2.2. The schema is motivated by a theoretical framework in cognition (expanded on below).
1.2.3. At a given "stop" on a subway line, the user is not forced to make a choice about where to go next; there is always a "default," namely, continue on the current subway line to the next stop. Only if the user wishes to follow a concept other than the one he or she is currently following must a decision be made.

2. Description of the basic design

2.1. The data structure and visual cohesion

Suppose that there is a large amount of material that is to be computerized and made accessible for browsing. The material is first divided into units (sometimes called frames or pages or screens), and each unit can be viewed as a node in a graph. The nodes are connected together by links, and a person using the system can traverse the material by traveling along the links. So a link represents the possibility of going from one node to another.

A determination of which nodes are linked together is made as follows. In each node, there is a set of designated elements. A link l between node i and node j is a relation between a unique element x in node i and a unique element y in node j. The link means that x and y are cohesive elements. For example, node i might contain a picture of a motor, and node j the word "motor." The picture and the matching verbal label refer to the same concept, and so can be designated as cohesive elements.

To give a brief background, the concepts of text cohesion and coherence have been used in similar ways by many authors (Halliday, 1985; Halliday & Hasan, 1976; Grimes, 1975; Harris, 1952; Kintsch & vanDijk, 1978). Cohesion is a semantic relation between two items in a text. For Halliday & Hasan (1976) cohesion occurs through word repetition, a noun and its pronoun referent, use of synonyms, etc. According to these authors, the semantic continuity provided by cohesion is a primary factor in a text's intelligibility.

Baggett & Ehrenfeucht (1982) extended text cohesion notions to visual, and to text and visual, or between-media cohesion. They described a "cohesion graph" for 23 frames (still photos) taken from an animated movie of James Thurber's The Unicorn in the Garden. Ten of the photos (which were drawn for us by Pam Hoge), and the cohesion graph created from all 23 arranged in the order in which they occurred in the movie, are shown in Figures 1 and 2. The cohesion graph was formed as follows. Nineteen concepts whose referents occurred in the photos, such as characters (e.g., husband, unicorn, wife) and specific locations (garden, bedroom, etc.), were selected. If a referent to any of the 19 concepts (i.e., a cohesive element) occurred in a photo, a mark was made in the cohesion graph, as shown in Figure 2. For example, it was determined that a picture of the husband occurred in 14 of the 23 photos. In the cohesion graph, the number of times cohesive elements occurred in two adjacent photos (when the photos were put in the order in which they occurred in the movie) was indicated. For the picture of the husband, this number was 11. The sum of these adjacencies was defined as

Figure 1. Ten of the 23 photos used in the Unicorn in the Garden ordering task. Numbers in upper left corners indicate the position in the actual story.

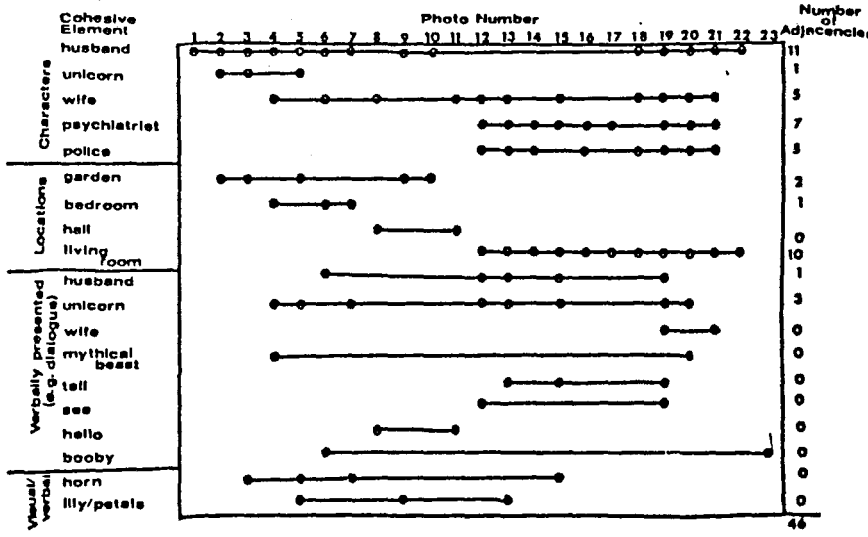


Figure 2. Cohesion graph of movie photos in correct order, as they occur in the actual story. Markers indicate that a particular cohesive element occurs in a given photo. The number of times a given element occurs in two adjacent photos is recorded in the far right column under "Number of Adjacencies."

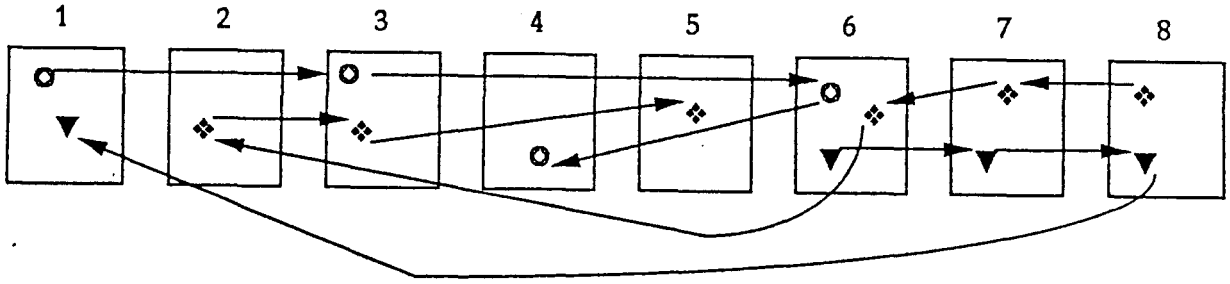
the cohesion value for the photos. The cohesion value for the 23 photos in their correct order was 46. One experimental result from the study was that people who had not seen the movie, when asked to put the 23 frames in an order so that they "make a good story," gave orders which on average had a cohesion value of 55.1, or nearly 20% greater than in the actual story. One interpretation for this result was that associations in memory may be made by an awareness of the existence of similar elements (e.g., as postulated by the 1978 theory of Kintsch et al.).

Baggett, Ehrenfeucht, & Guzdial (1989) developed a prototype interactive graphics-based instructional system for assembly and repair, in which access was via visually (or graphically) cohesive elements. The graphics implementation, together with prototypes in literature analysis, a history of mathematics, and elementary mathematics lesson plans, will be briefly described in section 3 below.

2.2. An example of a graph and its traversal

Figure 3 shows a hypothetical small example of a graph created from eight units of information. (In actuality, of course, graphs are considerably larger.) There are 3 types of cohesive elements in the figure, ▼, ✪, and ❖. Note that the path linking together cohesive elements of a given type does not branch, but forms a single line. In a given frame, the user can select, for a specific cohesive element, the "next" or the "previous" frame that contains that element, or the "home" (or first) frame that contains the element. For example, in frame 3, where there is a second occurrence of ✪. selecting "next" for ✪ takes one to frame 6; selecting "previous" takes one to frame 1, and selecting "home" takes one to frame 1. Similarly, in frame 3, selecting "next" for ❖ gives frame 5; "previous" gives frame 2, and "home" gives frame 8.

Although it was not done in Figure 3, "frame" could also be designated as a type of cohesive element. When this happens, "frame" acts as a page turner. By selecting "next" for frame, the user would go from frame 1 to frame 2, etc.



The physical order is indicated by 1, 2, 3, ...
There are three different cohesive elements, ✪, ❖, and ▼.

Figure 3. A graph of eight units of information and three types of cohesive elements.

2.3. Options available to the user

The following options can be made available to the user in every frame: For any cohesive element in the frame, the user may be able to ask about (1) the name of the cohesive element; (2) the number of occurrences of the element in the entire material; and (3) the number of the occurrence in the frame the user is currently visiting. So for the cohesive element "page number," the user could find out how many pages (or units) of information there are in the whole presentation, and which one is currently being visited.

In Figure 3 the user might be in frame 6 and ask about cohesive element ✪. Suppose the presentation is about an object to be repaired. The user might be informed that the name of the cohesive element is motor. He or she can also find out that there are four occurrences of motor as a cohesive element, and that the current occurrence is number three. The actual instances of "motor" might be moving video showing what happens when the motor is turned on, a cut-away diagram showing the interior of the motor, a text telling how to troubleshoot the object when the motor is dead, a list of parts in the motor, together with their photographs, etc.

The point is that all the instances refer to the same concept. A user following a cohesive element will receive information on the same concept, but possibly in different modalities. As noted above, a path linking together cohesive elements of a given type does not branch. The order of access is determined by the designer. Also, not every reference to the concept (e.g., motor) has to be designated a cohesive element. For example, if the reference is of minor importance in a frame, the designer may decide to omit it as a cohesive element in the frame.

To help the user keep from getting lost, a stack facility can be provided, so he or she can always go back to previously visited material.

Other options available to users can be quite elaborate. For example, a user may be able to request, "Take me to the ninth stop on this line," or "Take me to the stop on this line that intersects with such-and-such other line."

2.4. Responsibilities of the designer.

To design a presentation, a person must do five things:

1. Select a set of materials to be browsed.
2. Divide the material into units, which will be nodes in the graph.
3. Designate one or more parts of each unit a cohesive elements of a given type.
4. Give each cohesive element type (each subway line) a unique short name, which will be available to a user.
5. Specify, for each element in each unit, what its predecessor and its successor will be, in terms of access.

2.5. Graph theoretical results that indicate this should be a good approach

It is desirable that any two nodes are connected by a short path. (It may be that the user needs to change subway lines in order to find the short path.)

There is a theorem (Bollobas, 1985) which states that if a graph is created randomly, then the expected distance between two nodes (i.e., the length of a shortest path between them) is almost as small as is theoretically possible. (The theoretical minimum depends mainly on the valency of the graph, namely, the number of nodes that can be linked directly to one node). This result indicates that the designer does not have to worry about short connections between nodes. This property is expected to occur by itself.

3. Descriptions of implementations

In this section we briefly describe some implementations of the design. All were done on the Macintosh and are mouse-driven.

3.1. Graphics-based procedural instructions for repair of a 'string crawler'

Figures 4, 5, 6, and 7 show the first four frames from an interactive graphics-based instructional system for assembly and repair, in which access was via visually (or graphically) cohesive elements (Baggett et al., 1989). The object in question was a "string crawler," a battery-powered 'vehicle' which travelled along a string when its switch was turned on. A user of the system could follow any of 21 different concepts (e.g., motor, batteries, wire, switch box) through the 41-frame presentation when given a task (to repair a broken string crawler). Cohesive elements were indicated by stars (see Figures 4 through 7); clicking on a star or its related object moved the user to the 'next' frame in the presentation that contained the object. There were 98 clickable stars in the presentation and thus an average of about 2.4 per frame.

3.2. Emily Dickinson poems

Two frames from another implementation of the design, by the first author, are shown in Figures 8 and 9. They involve 183 Emily Dickenson poems and critiques of them offered by four authors. The cohesive elements in this case are 16 abstract themes, together with the poems themselves. In Figures 8 and 9, a poem subway line, [P 303] The Soul selects her own Society, is shown. (The subway line is indicated below the banded line on the lower left-hand side of the screen.) The user is on stop 2 of 5 for this poem. Stop 1 was the poem itself, while stop 2 is the poem together with a critique of it by Charles R. Anderson. The sentence beginning "However fleeting love's glory..." is highlighted because it is here in Anderson's critique that his discussion of The Soul selects her own Society begins. There are three other cohesive elements (subway lines) active in this frame; two are thematic: Love Imagery and Sexuality Imagery. And the third is the poem [P 249] Wild Nights - Wild Nights (Anderson is also discussing it in this frame). The user can get the menu of concepts, with those that are active in this frame blackened, by clicking on the compass icon. By clicking on any of the three active elements in the menu, the user can switch subway lines.

While the Emily Dickenson implementation is not multimedia, it is based on more than text matching. Its cohesive elements for the most part refer to abstract themes.

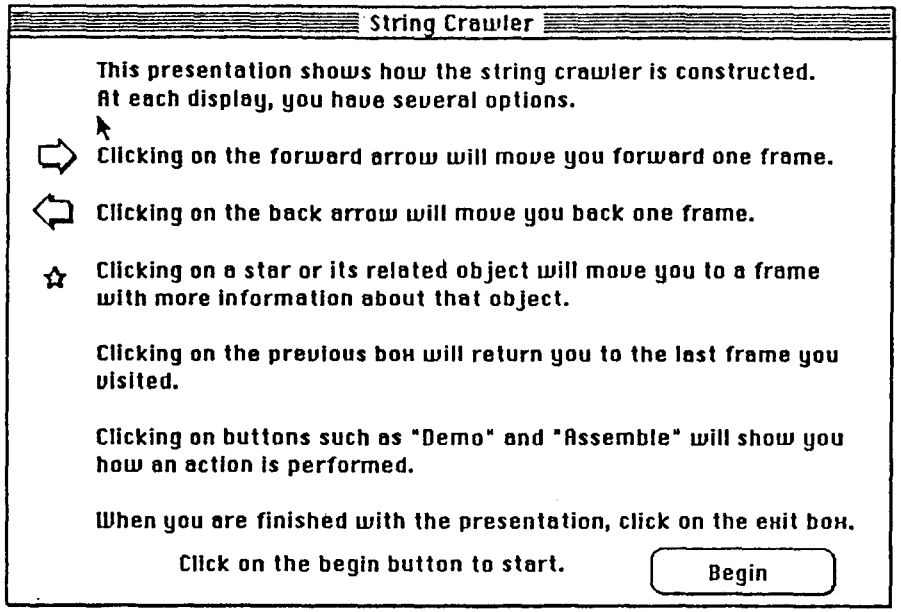


Figure 4

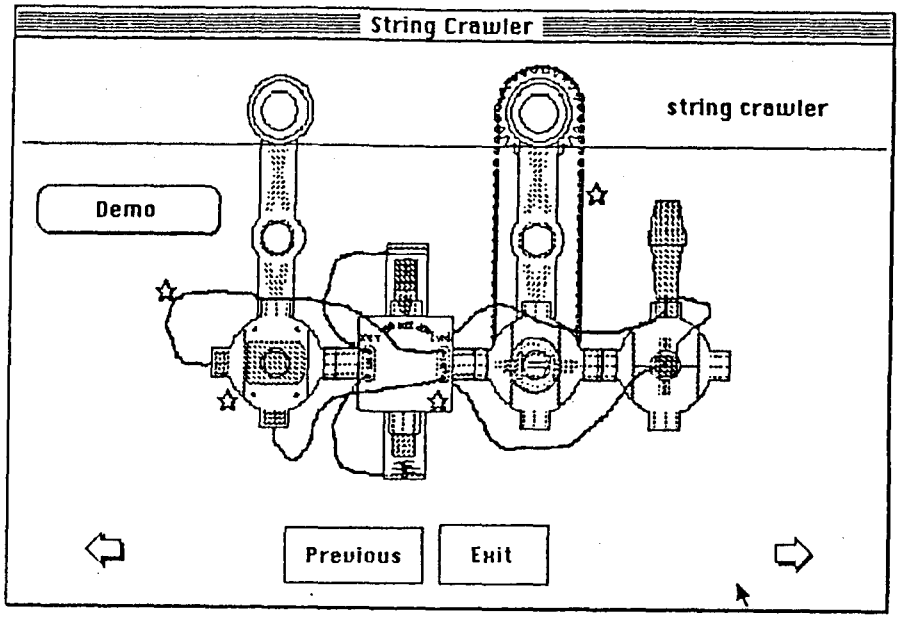


Figure 5

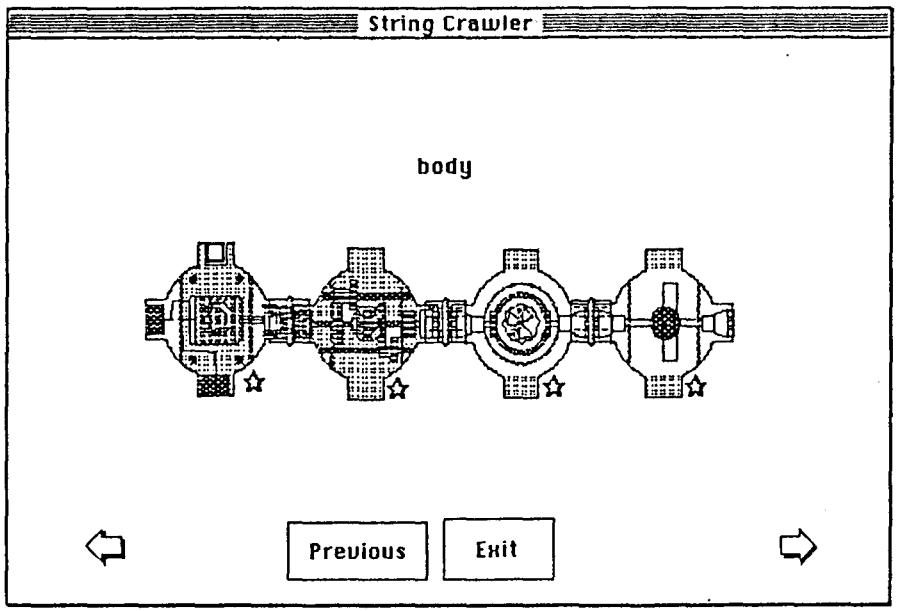


Figure 6

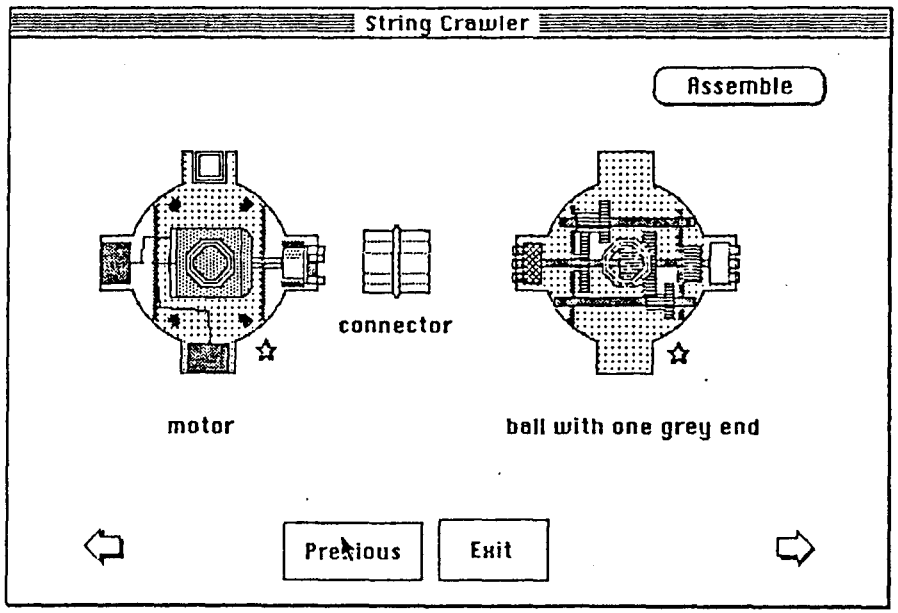


Figure 7

Figures 4, 5, 6, and 7. The first four frames from an interactive graphics-based instructional system. Cohesive elements are indicated by stars.

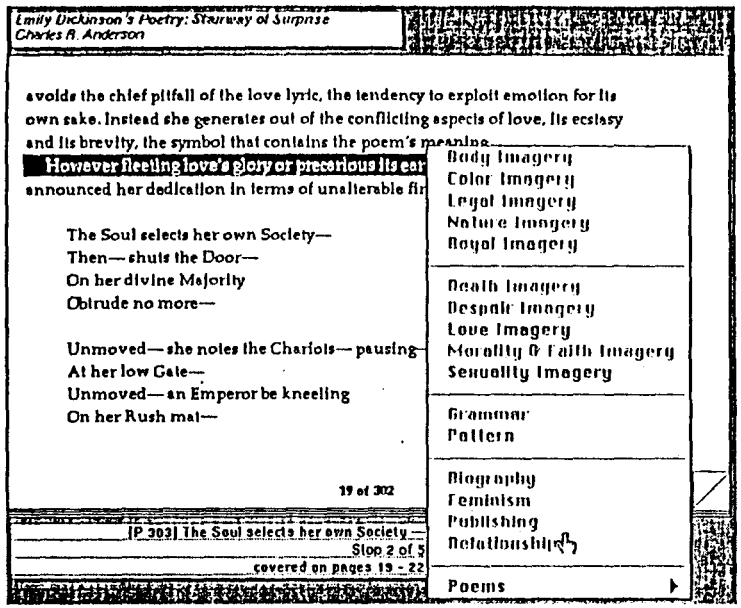


Figure 8

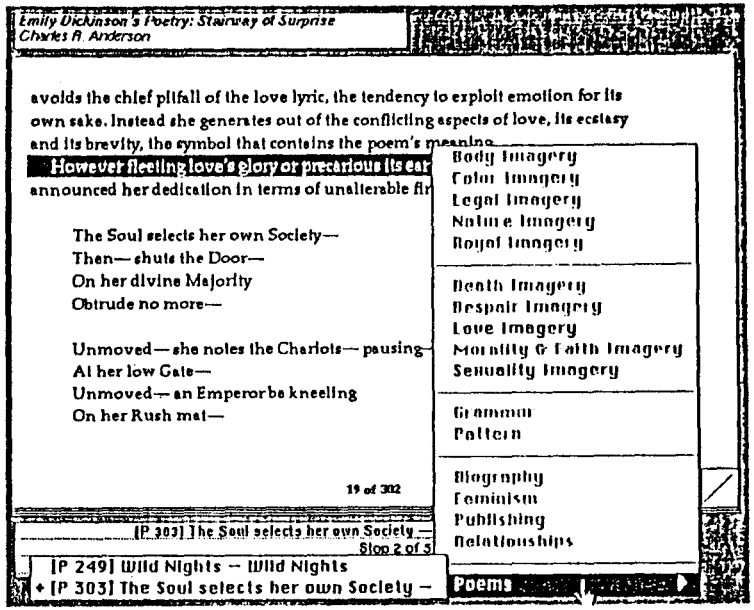
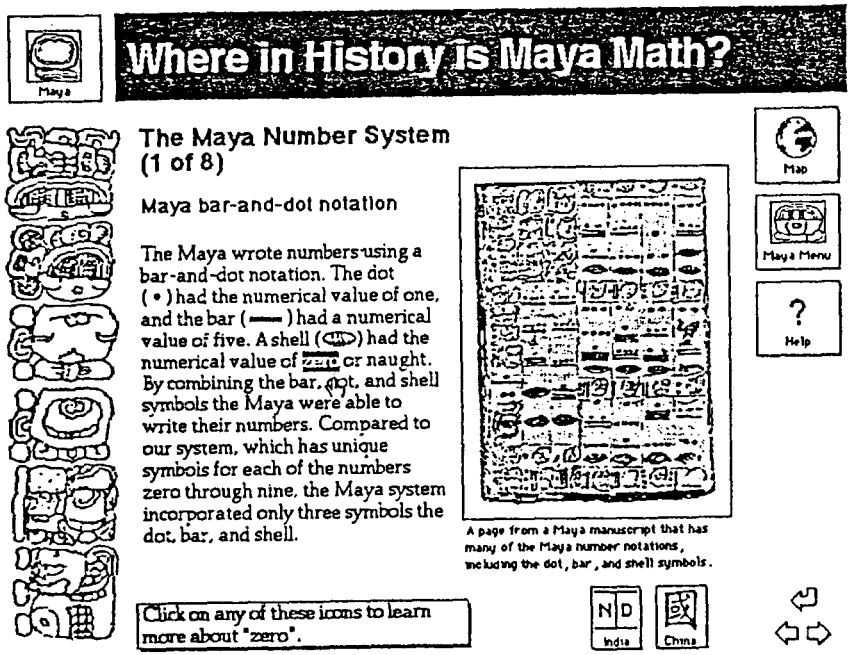
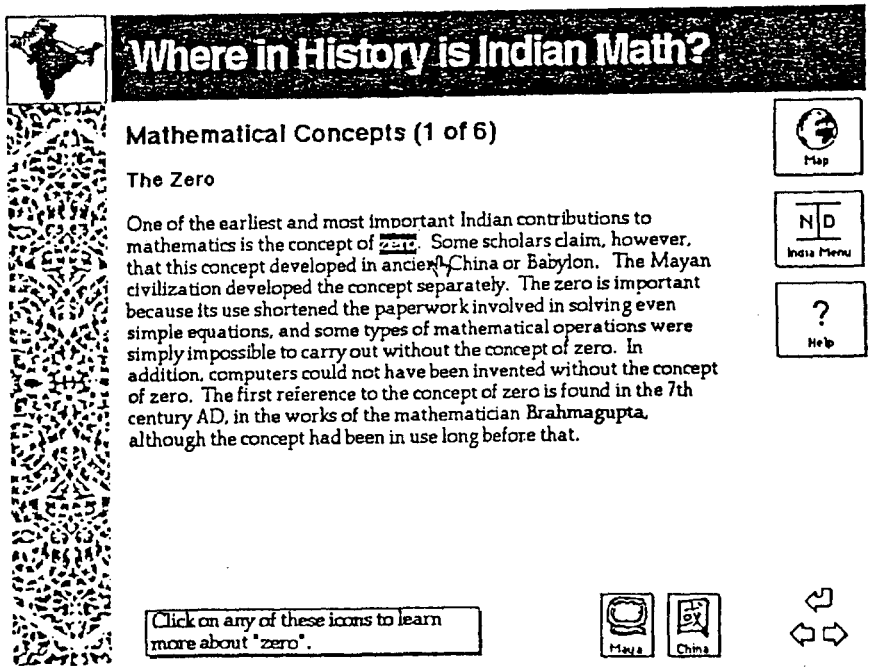


Figure 9

Figures 8 and 9. Frames from an implementation involving the poems of Emily Dickenson.





3.3. Where in history is mathematics?

This implementation, done by students in two graduate software design classes taught by the second author, involves the six different civilizations which according to current knowledge developed mathematics at least in part independently: Africa, Sumer-Babylon, India, China, Maya, and Inca. Concepts whose referents make up the cohesive element list include zero, counting schemes, tools, games, and famous individuals. Figures 10 and 11 show two frames, from the Mayan and Indian civilizations, linked by references to the concept zero.

The implementation was originally laid out as a two-dimensional array. Names of the six civilizations labeled the columns, and each row was labeled by a concept. In theory, if all civilizations had references to each concept, the user could travel through the matrix horizontally or vertically. In reality, however, it was not the case that a reference to each concept was included for each civilization. For example, a reference to the concept of zero was included only in the civilizations of Maya, India, and China.

The implementation has been updated and extended over two semesters, and it is still in progress. The most recent addition is the concept of mathematical games in some of the civilizations.

3.4. Elementary mathematics lesson plans

Figures 12, 13, and 14 show frames from a prototype implementation involving a database of 101 lesson units from a new elementary mathematics curriculum. The implementation was done by Barry Webster under the supervision of the second author, using materials written by Baggett & Ehrenfeucht (1993). The materials are meant for teachers. Figure 12 shows the 16 concepts which can be followed through the materials in the prototype, and Figures 13 and 14 show two frames linked by the concept "area." The plan is for this prototype to be extended to a larger database (currently about 250 lesson units have been written); concepts yielding subway lines will be chosen from typical "scope and sequence" charts found in elementary textbooks. Teachers will have the materials available electronically, and if they want to browse all units which contain a certain concept (e.g., place value), they can do so. Many lesson units contain illustrations, so that this implementation will as it expands have a multimedia database.

4. Advantages of this approach

4.1. Our work (Baggett & Ehrenfeucht, 1988; Baggett, Ehrenfeucht, & Guzdial, 1989) has shown that when material is presented passively to a learner (e.g., as in a videotape), the hierarchical structure of the material plays an important role in learning. But when the information is presented interactively, so that a user can traverse the material freely, cohesive elements are more important than hierarchical structure. This design gives principled access to information based on one's being able to follow a line of interest.

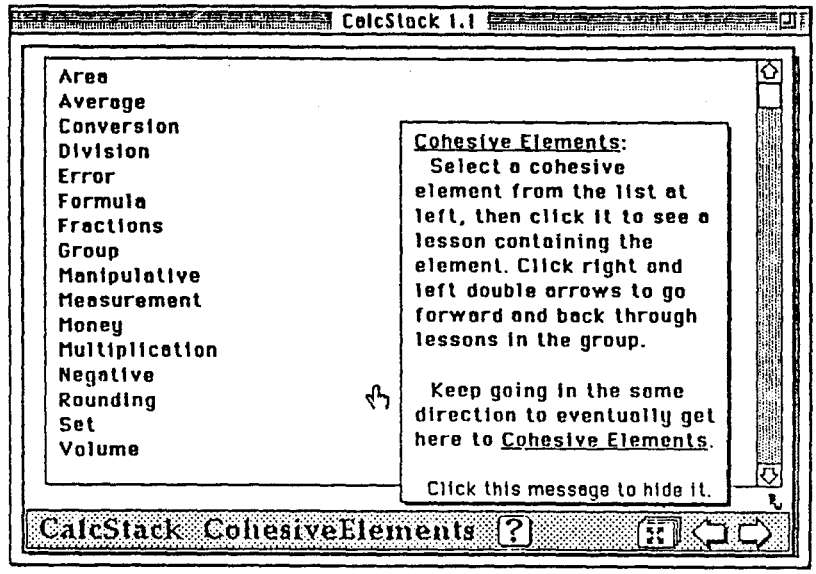


Figure 12

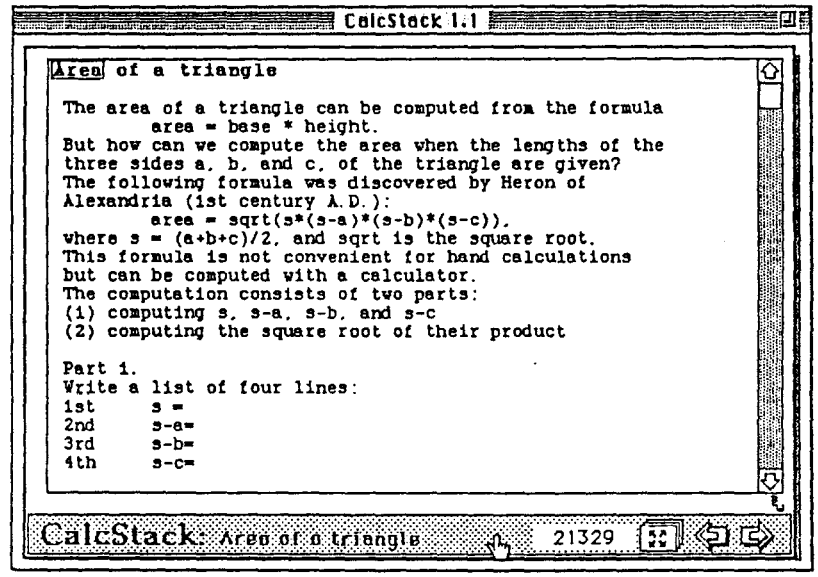


Figure 13

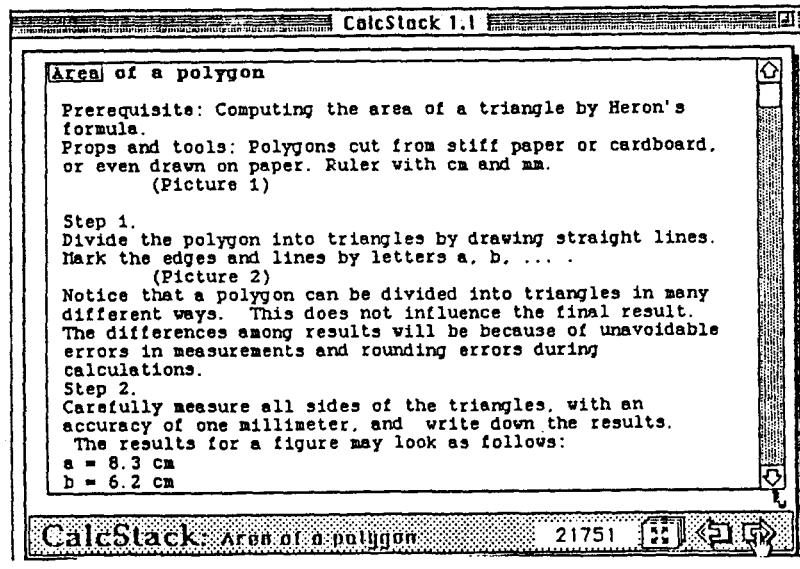


Figure 14

Figures 12, 13, and 14. Frames from a prototype implementation involving lesson plans from an elementary mathematics curriculum.

4.2. Since individuals are allowed to follow their own "tours" through lengthy material, rather than having to take "guided tours," they may take more active roles in exploring the material and get directly to the information that they want, quickly, without confusion, and without wading through irrelevant and distracting material.

4.3. On every frame the information that is available to the user about cohesive elements can be implemented to act as a constant index. This on-line index helps him or her follow a line of inquiry, quickly spot old and new important topics in a unit, and be kept aware of where in the information he or she currently is. This type of information is not available in a book. It should give users a good "feel" for the material, and help them locate what they want efficiently.

4.4. The system allows the designer to bypass having to find the hierarchical structure of material to be presented. It is based on retrieval of information by concepts designated as important by the designer, and independent of the medium in which they are presented. Users explore and learn by browsing and following their own choices of concepts through the data.

There have been attempts at developing on-line browsing systems in recent years, and electronic access to large bodies of material is the wave of the future. The novelty of our approach lies in arranging access so the users/learners can follow single cohesive lines through the material. The ideas are simple, and theoretical results in random graph theory indicate that the method can result in high efficiency for users in exploring. locating, and studying information.

References

Baggett, P. & Ehrenfeucht, A. (1982). Information in content equivalent movie and text stories. Discourse Processes, vol. 5, 73-99.

Baggett, P. & Ehrenfeucht, A. (1988). Conceptualizing in assembly tasks. Human Factors, 30 (3), 269-284.

Baggett, P. & Ehrenfeucht, A. (1993). A new elementary mathematics curriculum that incorporates calculators. Unpublished materials.

Baggett, P., Ehrenfeucht, A., & Guzdial, M. (1989). Sequencing and access in interactive graphics-based procedural instructions. University of Michigan School of Education Technical Report for Office of Naval Research, vol. 2 (1), August.

Bollobas, B. (1985) . Random Graphs. New York: Academic Press.

Grimes, J. (1975). The Thread of Discourse. Mouton Press, The Hague, 1075,

Halliday, M. (1985). An Introduction to Functional Grammar. Baltimore, MD: Edward Arnold.

Halliday, M. & Hasan, R. (1976). Cohesion in English. London: Longman Press.

Harris, Z. (1950). Discourse Analysis. Language, 28, 1-30.

Kintsch, W. & vanDijk, T. (1978). Toward a model of text comprehension and production. Psychological Review 85, 363-94.

"Class, you're not making enough noise!"
The Case for Sound-Effects in Educational Software

Bill Gaver

Rank Xerox Cambridge EuroPARC
61 Regent Street, Cambridge CB2 1AB, UK

1 Introduction

In this paper, I argue that educational software could benefit in a number of ways by the incorporation of sound-effects much like those used in movies, radio dramas, and video games. Classes should be filled with the sounds of roaring machines, running water, even breaking glass – not made by the real thing, of course, but by educational software.

If this seems crazy, consider KidPix™, an award-winning drawing program for children produced by Bröderbund Software (see Figure 1). KidPix is fun: It incorporates a large number of innovative graphical tools with which kids can explore and play. This figure shows what a few of them do. Pick one paint brush and drops of ink follow the cursor. Pick another and trees grow wherever you press the mouse button. Yet another turns smooth mouse motions into rough scribbles. Use one of the "rubber stamps" to put little drawings wherever you click. And if you don't like the result, you can erase it in a number of highly satisfying ways – my favourite is the bomb.

KidPix is an interesting program in part because it uses the power of the com-

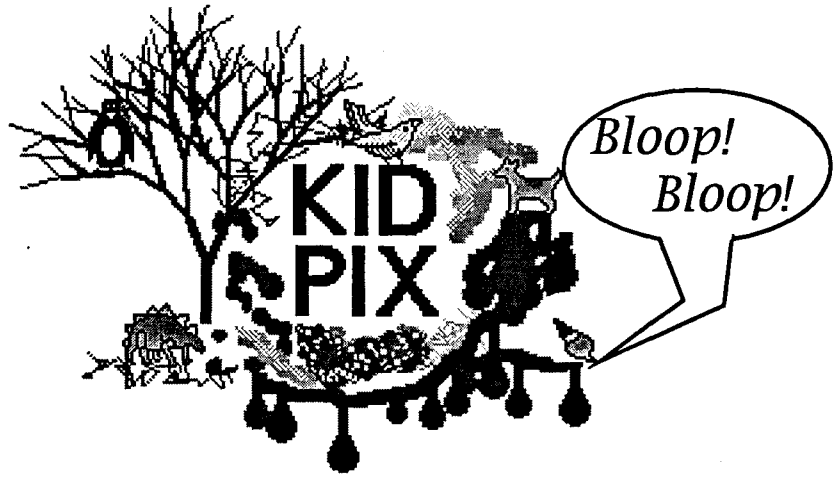


Fig. 1. KidPix™ uses a variety of sound-effects to accompany drawing.

puter to go beyond the usual literal metaphors used in painting programs. But it is the *sound* that makes it more than just a novel drawing program: all the tools are accompanied by clever, useful, and appropriate sound-effects. Use the dripping paintbrush, for instance, and you hear the drops of paint seep down the screen. Use the scribbling paintbrush, and you hear a pencil scribbling on paper. Stamp an owl on the tree branch, and you hear the stamp slap against the page. And when you blow up the screen, you hear the explosion – the effect wouldn't be nearly so powerful otherwise.

Are these sounds just gimmicks? Perhaps. But I argue that the sounds used in KidPix add significantly to its success in a number of ways, and that these advantages are applicable to many other programs.

1.1 What Sounds Add to Interaction

First, sounds increase the *tangibility* of the interface, and help to explain what is happening. Remember that the users of KidPix are unlikely to trouble with manuals or help systems – if they can read at all. The sounds help to make the program self-explanatory. Consider how confusing the scribbling paintbrush would be if it wasn't accompanied by the appropriate noise. Or how prosaic the bomb-eraser would be if it "blew up" your screen without making a sound. Sounds like these reinforce the visual expressions of a metaphor. By adding redundant information about what is happening, they make the tools seem more real (and thus the interface more transparent). Because they also add new information – for instance, about the putative tool and surface, in the case of the scribbling paintbrush – they can convey new information about what is going on in the interface.

Second, sounds can help *coordination* with others, by allowing other kids (and teachers) to hear what they can't see. You don't need to be looking at the screen to know what tools a kid is using – you can hear the scribbling, dripping, and explosions. This may not be so important for programs like KidPix, but it is potentially quite useful for educational software, particularly collaborative systems in which several people may be working in the same program at the same time.

Finally, sounds promote *guided exploration*, by selectively indicating attributes of interest. Real sounds inevitably accompany actions, determined by the laws of physics. Sound-effects have to be created; programs do not naturally make sounds. This means that the designers of sound-effects have the opportunity to design sounds that highlight some aspects of an event and not others (in fact, they can't avoid it). So the scribbling paintbrush stresses the gestures of pencil on paper, while the dripping brush leaves out the surface, and emphasises the viscosity of the "ink." Again, this may not be so important for entertainment software like KidPix, but it can be valuable for educational software.

2 Auditory Icons

The sounds used in KidPix™ were designed to be fun, not useful. It is doubtful that the designers sat down and asked themselves, "how can the sounds we use increase tangibility, promote coordination, and help with guided exploration?" They were, after all, primarily concerned with making their program entertaining and useful enough to sell. Thus while most of the sounds in this program are remarkably apt, few are actually that useful.

Over the last several years, on the other hand, I have been concerned with just these issues. The result is a strategy for creating *auditory icons*, everyday sounds designed to convey information about computer events by analogy with everyday events. The sounds I use are similar to those used in KidPix. But because the purpose of auditory icons is to provide information, rather than entertainment, a richer, more principled set of design principles have evolved.

The basic idea behind auditory icons is to map computer events to analogous sound-producing events (see Figure 2). For instance, when a user selects a file, the act of clicking the mouse while the cursor is positioned over the icon suggests that a tapping noise might be an appropriate mapping. The result is a simple sound-effect, much like those used in KidPix.

But auditory icons go beyond the sounds used in KidPix because they are *parameterized*; that is, meaningful parameters of the computer and sound-producing events are mapped to one another. For instance, the size of the file might be indicated by the size of the virtual object you hear tapped. The type of file can be conveyed by the material of the tapped object. And the overall disk space might be mapped to the overall reverberation applied to the sound, so that an empty disk sounds like a large, echoing hall, and a full disk sounds liked a small, cramped room. Parameterizing auditory icons has two effects: First, it allows families of auditory icons to be created, which convey rich information in lawful ways. Second, it increases the variability of the sounds heard from the interface, and thus helps prevent any given sound from becoming annoying through repetition.

Auditory icons rely on a new approach to the psychology of sound and hearing that stresses our tendency to hear sounds in terms of the events that cause them (see Gaver, 1993a, 1993b). That is, if you are walking down a street and hear a sound, you are less likely to ponder its pitch, loudness, and timbre (all attributes of sound studied by traditional psychophysics) and more likely to hear it as a large automobile heading your way – and jump! This experience of *everyday listening* implies that we can start to describe sounds and their perceptual correlates in terms of events. Instead of describing a sound in terms of its pitch and the acoustic correlate of frequency, for example, we might describe it in terms of the size of its source and the associated acoustic correlates. This perspective leads to a new

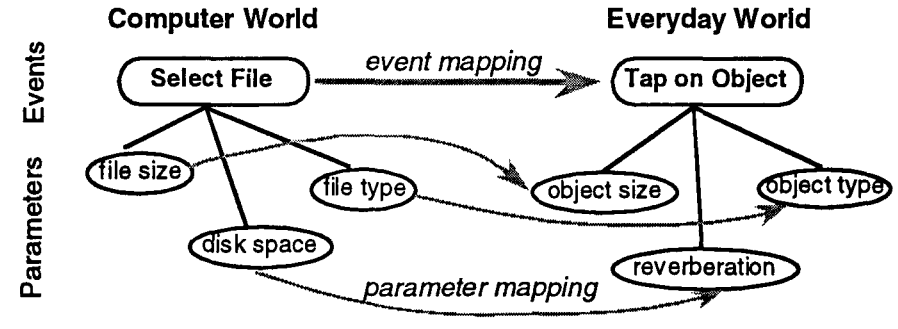


Fig. 2. Auditory icons are created by mapping events in the computer world to sound-producing events in the everyday world. They may be parameterized by mapping attributes of the computer and sound-producing events to one another.

approach to understanding the psychology of sound and hearing, as I point out above. More important, for the purpose of this paper, it leads to a new set of conceptual tools for building auditory interfaces: We can now think mapping events in the computer directly to sound-producing events. This is the strategy behind auditory icons.

In the rest of this paper, I describe auditory icons and their potential uses in educational software. My discussion centers around the three functions of increasing tangibility, supporting coordination, and guiding exploration. Each of these functions is illustrated by a system that uses auditory icons. The first is the SonicFinder, an auditory extension to the desktop metaphor which makes the world of the computer more tangible and more self-explanatory. The second is SoundShark, and a related application ARKola, which adds sound to a prototype system of collaborative software meant to support distance education. The last is the Environmental Audio Reminder (EAR) system, which uses sound to convey selected information about events going on in our office environment.

3 Increasing Tangibility: The SonicFinder

Educational software must achieve a number of goals. The primary one, of course, is to teach students about the subject matter at hand. But to do this, the students must learn to use the software itself. Thus a second goal is that the software should be readily learned and ideally self-explanatory. In addition, software should motivate students both to explore the particular contents it offers and the relevant domain more generally. This is a third challenge facing designers of educational software.

Auditory icons may help in creating interfaces that are easy to learn and which motivate users by increasing the tangibility of the interface. A good example of this is the SonicFinder (Gaver, 1989). This is an extension to the Finder, the application used to organise, manipulate, create and delete files on the Macintosh. Creating the SonicFinder required extending the Finder code at appropriate points to play sampled sounds modified according to attributes of the relevant events. Thus a variety of actions make sound in the SonicFinder: selecting, dragging, and copying files; opening and closing folders; selecting, scrolling, and resizing windows; and dropping files into and emptying the wastebasket. Most of these sounds are parameterized, although the ability to modify sounds was limited by the sampling software used. So, for instance, sounds which involve objects such as files or folders not only indicate basic events such as selection or copying, but also the object's types and sizes via the material and size of the virtual sound-producing objects. In addition, the SonicFinder incorporates an early example of an *auditory process monitor* in the form of a pouring sound that accompanied copying and that indicates, via changes of pitch, the percentage of copying that had been completed (c.f. Cohen, 1993).

Figure 3 shows an example of an interaction with the SonicFinder. When a user selects a file icon (Figure 3A), a tapping sound is played which provides feedback about the event and also provides information about the file size and type. As the user drags the icon towards the wastebasket, a scraping sound is played (3b), which not only indicates that the object is being moved, but also reflects the size of the file and what it is being dragged over (i.e., its home window, another window, or the desktop). This sound stops either when the user releases the object, or when it

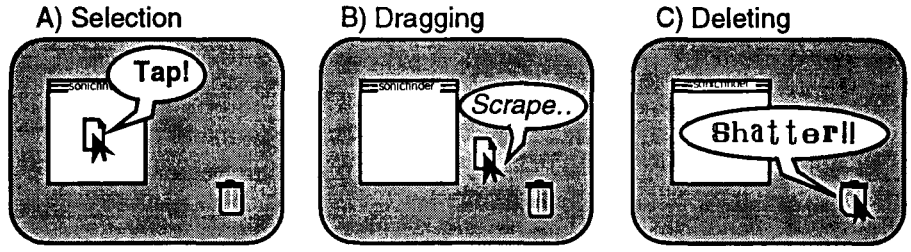


Fig. 3: In the SonicFinder, many user-initiated events are accompanied by meaningful auditory icons, increasing tangibility and ease of learning.

hits a possible container such as a folder. Finally, when the user throws the icon into the wastebasket (3C), marking it for deletion, the sound of crashing glass is heard. This provides useful feedback about the nature of the interaction, and also reflects the number of other objects in the wastebasket.

Sounds like these serve to provide redundant information about events in the system, and thus increase its tangibility. Once users become accustomed to the SonicFinder, they feel somehow removed from the interface if the sound is turned off. The sounds seemed to make interactions more immediate to users.

Sounds also convey information that is not conveyed graphically. For instance, the size and type of files is not indicated in most modes of the graphical display, yet can be readily judged (at least qualitatively) from the selection sounds. Sometimes sounds seem to convey information more effectively than graphic feedback. For example, hearing when a dragged object is over a container seems more effective than highlighting for helping people avoid playing chase-the-wastebasket, and in moving files to new windows without losing them in unnoticed folders. Finally, there is some anecdotal evidence that the SonicFinder helped some new users understand the underlying metaphor of the Finder; by emphasising the tangibility of icons, they support the interpretation of simple line-drawings as "objects" that can be moved, opened, etc.

In sum, the SonicFinder demonstrates the potential for auditory icons to provide rich information in intuitive ways. This can be helpful in making systems obvious, and thus in allowing users – whether students or not – to focus on the content offered by the software rather than the task of figuring out how to use it. In addition, auditory icons can increase the feeling of *direct engagement* (Hutchins, Hollan & Norman, 1986) with interfaces. They reinforce the feeling that the model world presented by graphical interfaces is a real one that can be interacted with in meaningful ways. Finally, interfaces that use auditory interfaces are more fun: Just as erasing the screen in KidPix would be boring if the bomb were silent, so selecting a file in the Finder is somehow less satisfying when it doesn't make a noise.

4 Supporting Coordination: SoundShark and ARKola

One of the potential problems with using educational software is that it may separate students from each other and from their teachers, turning classrooms into collections of individual users, each staring intently into his or her own screen. Thus another challenge for the designers of educational software is to develop prod-

ucts that support communication and coordination amongst students and their teachers. Auditory icons can help with this both within the physical classroom, and within collaborative virtual worlds used as "classrooms" for distance education.

4.1 SoundShark

An example of this latter role for sound is provided by SoundShark, an auditory version of SharedARK (Smith, 1989). SharedARK is a collaborative version of ARK, the Alternate Reality Kit. Developed by Smith (1987), ARK is designed as a virtual physics laboratory for distance education. The "world" appears on the screen as a flat surface on which a number of 2.5D objects may be found. These objects may be picked up, carried, and even thrown using a mouse-controlled "hand." They may be linked to one another, and messages may be passed to them using "buttons." Using this system, a number of simple physical experiments may be performed. In addition, SharedARK allows the same world to be seen by a number of different people on their own computer screens (and is usually used in conjunction with audio and video links that allow them to see and talk to one another). They may see each other's hands, manipulate objects together, and thus collaborate within this virtual world.

SharedARK is a multiprocessing system, with the potential for several "machines" or self-sustaining processes to run simultaneously. In addition, it provides a very large world to users, in that the space for interaction is many times larger than the screen (depending on available memory, it may cover literally acres of virtual space). This means that simultaneous users of the system may not be able to see each other (or more accurately, the hands that represent them), despite their being in the same "world" and potentially changing it in ways that might affect each other.

To help collaboration in this large, complex world, we extended the SharedARK interface with auditory icons that indicate user interactions, provide background information about ongoing processes and modes, and support navigation. The result is called SoundShark (Gaver and Smith, 1990). Sounds are used to provide feedback as they were in the SonicFinder: Many user actions are accompanied by auditory icons which are parameterized to indicate attributes such as the size of relevant objects. In addition, ongoing processes make sounds that indicate their nature and continuing activity even if they are not visible on the screen. Modes of the system, such as the activation of "motion," which allows objects to move if they have a velocity, are indicated by low-volume, smooth background sounds.

These auditory icons are helpful for individual users. More interesting, the sounds support collaboration among remote users. Because each sound can be heard throughout the "world," collaborators can hear each other even if they can't see each other. In addition, the distance between a user's hand and the source of a sound is indicated by the sounds' amplitude and by low-pass filtering. This not only seems to aid navigation (prompting us to develop "auditory landmarks," objects whose soul function was to play a repetitive sound that could aid orientation) but allows collaborators to hear the distance and even direction of their partners.

4.1 The ARKola Simulation

Our experiences with SoundShark suggested that auditory icons could help collaboration amongst distributed colleagues interacting in a virtual environment. To test

this, we developed a special application within SoundShark that we used for observing people's use of the system. Our aim was to assess the usefulness of auditory icons for collaborators within the system as well as for individual users.

The application we came up with is a model of a softdrink plant called the ARKola bottling factory (Figure 4; Gaver et al, 1991). It consists of an assembly line of 9 machines which cook, bottle, and cap cola, provide supplies, and keep track of financing. The plant was designed to be fairly difficult to run, with the rates of the machines requiring fine tuning and with machines occasionally "breaking down," necessitating the use of a "repair" button. In addition, we designed the plant to be too large to fit on the computer screen, so participants could only see about half the machines at any given time.

Each of the machines makes sounds to indicate its function. For instance, the "nut dispenser" makes wooden impact sounds each time a nut is delivered to the cooker, the "heater" makes a whooshing flame-like sound, the "bottler" clangs and the "capper" clanks. In addition, the rate of each machine is indicated by the rate of repetition of the sounds it makes, and problems with the machines are indicated by a variety of alerting sounds such as breaking glass, overflowing liquid, and so forth.

As with SoundShark, the sounds were designed not only to be useful for individual users, but also to be helpful in coordinating partners running the plant. We tested this in a simple observational study. Six pairs of participants were asked to run the plant with the aim of making as much "money" as they could during an hour-long session. Each pair ran the plant for two hours, one with and one without auditory feedback (with the order, of course, being counterbalanced). We observed their performance from a "control room" via video links as they ran the plant, and videoed their activities for later analysis.

Our observations indicated that sounds were effective in helping individual users keep track of the many ongoing processes. The sounds allowed people to track the activity, rate, and functioning of normally running machines. Without sound,

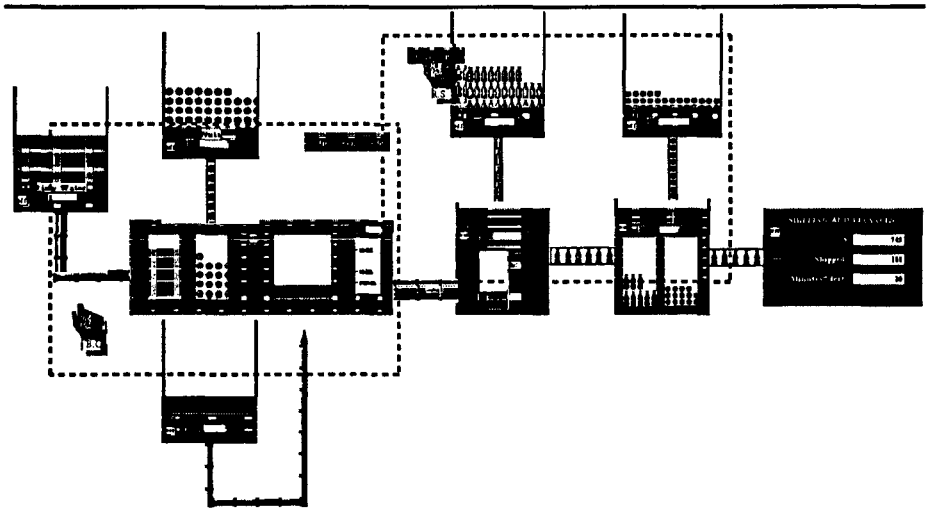


Fig. 4. The ARKola bottling plant simulation (about one fourth actual size). Rectangles show the extent of the plant each user sees at a given time.

people often overlooked machines that were broken or that were not receiving enough supplies; with sound these problems were indicated either by the machine's sound ceasing (which was often ineffective) or by the various alert sounds. Perhaps most interesting, the auditory icons allowed people to hear the plant as an integrated complex process. The sounds merged together to produce an auditory texture, much as the many sounds that make up the sound of an automobile do. Participants seemed to be sensitive to the overall texture of the factory sound, referring to "the factory" more often than they did without sound.

These observations support the idea, introduced in Section 3, that well-designed sounds can help new users learn how a system works. These observations also support the thesis of Section 4, that sounds are particularly useful in guiding exploration of systems by providing information about a relevant subset of events. Finally, sound seemed to add to the tangibility of the plant and increased participants' engagement with the task. This became most evident with a pair of participants who had completed an hour with sound and were working an hour without. From the video, their increasing boredom with the silent plant is obvious (as are several mistakes that they attribute to the lack of sound). Finally, one of the pair remarks "we could always make the noises ourselves..."

Our major findings, however, related to the role of sound in collaboration. In both the sound and no-sound conditions, participants tended to divide responsibility for the plant so that each could keep one area on the screen at all times. Without sound, this meant that a each had to rely on their partner's reports to tell what was happening in the invisible part. With sound, each could hear directly the status of the remote half of the plant. This led to greater collaboration between partners, with each pointing out problems to the other, discussing problems, and so forth. The ability to provide foreground information visually and background information using sound seemed to allow people to concentrate on their own tasks, while coordinating with their partners about theirs. It was an effective way of linking participants in this model world without forcing them to be together all the time. This is likely to be a very useful feature for helping with the balance between self-guided work and coordination in educational systems of the future.

5 Guiding Exploration: EAR

Still another challenge facing designers of educational software is in balancing between tightly-constrained teaching systems and systems that allow free exploration on the part of students. Traditional "drill and test" systems seem boring, have dubious efficacy, and make limited use of the potential power that computing offers. Newer "exploratory learning" systems, on the other hand, may be confusing and, moreover, make it difficult to coordinate the material that students actually learn as they wander through complex virtual environments.

A middle ground between these extremes may be found in systems which support *guided exploration* of complex environments. The idea is to constrain students' explorations of rich informational spaces to a set of trajectories that cover a subset of relevant information. For instance, fictional characters have been used to guide access to a historical database using the conventions of narrative flow (Laurel et al., 1990). Systems such as these offer a good deal of flexibility to teachers: Students may be assigned one particular trajectory, permitted to chose amongst a number of alternatives, or allowed simply to wander freely through the information space.

Auditory icons can help create systems that offer guided exploration by allowing a subset of information to be presented at a given time. From this point of view, sounds can be designed not to convey all possible information, but to present that which is necessary to highlight a particular task. For instance in SoundShark different sounds might be used for lessons involving gravity (for which mass, distance and the like are important) than for those involving the coordination of complex processes (like ARKola, in which the relative rates of the machines was most important). Using sounds, different aspects of the "world" can be emphasised for different purposes.

5.1 Environmental Audio Reminders

The key to using sounds that guide exploration is in designing them clearly represent only the relevant information for a given task, as opposed to all the information available. As an example of this, consider the Environmental Audio Reminders (EAR) system we use at EuroPARC (Gaver, 1991). This system plays a variety of nonspeech audio cues to offices and common areas inside EuroPARC to keep us informed about a variety of events around the building. EAR works in conjunction with the RAVE audio-video network (Gaver et al., 1992; Buxton & Moran, 1990), which connects all the offices at EuroPARC with audio and video technologies using a computer-controlled switch, and *Khronika* (Lövstrand, 1991), an event server which uses a database of events in conjunction with software daemons to inform us of a wide range of planned and spontaneous, electronic and professional events. EAR, then, consists of sounds triggered by Khronika when relevant events occur, which are routed using the RAVE system from a central server to any office in the building.

This system is set up to play sounds that remind us about a range of events. For instance, when new email arrives, the sound of a stack of papers falling on the floor is heard. When somebody connects to my video camera, the sound of an opening door is heard just before the connection is made, and the sound of a closing door just after the connection is broken. Ten minutes before a meeting, the sound of murmuring voices slowly increasing in number and volume is played to my office, then the sound of a gavel. And when we decide to call it a day, one of us may play the "pub call" to interested colleagues, who then hear laughing, chatting voices in the background with the sound of a pint glass being filled with real ale in the foreground.

Many of the sounds we use in EAR may seem frivolous because they are cartoon-like stereotypes of naturally-occurring sounds. But it is precisely *because* they are stereotyped sounds that they are effective. More "serious" sounds – such as electronic beeps or sequences of tones – would be likely to be less easily remembered than these. In addition, we have taken some care in shaping the sounds to be unobtrusive. For instance, many of the sounds are very short; those that are longer have a relatively slow attack so that they enter the auditory ambience of the office subtly. Most of the sounds have relatively little high-frequency energy, and we try to avoid extremely noisy or abrupt sounds. So though the sounds we use are stereotypes, they are designed to fit into the existing office ambience rather than intruding upon it.

EAR serves as a useful example of many of the themes of this paper: They are effective because they are stereotyped sounds that are easily learned and remembered.

They promote coordination by making information available to the distributed members of our labs. The overall effect is to allow us to hear events as if they were just outside our offices, despite the fact that these events are either too distant to hear naturally or have not even occurred yet.

The point here, however, is that while EAR creates an intuitively accessible auditory environment that complements and supplements our everyday one, it is not a slavish imitation of *all* the sounds we might potentially hear around our building. We don't play the sounds of people entering and leaving the building, for instance, or those of people typing, moving things in their offices, etc., though in principle we could. These events do not appear relevant to the tasks that we are supporting, and thus would merely be distracting. Instead we choose the sounds we play to indicate an important subset of the ongoing activities in the building. Moreover, individual users may tailor the system, registering interest in some events (e.g. meetings), but not others (e.g., pub time). In this way, the system supports a guided trajectory through the information that might be available. It offers the ability to maintain awareness of remote events without distracting or confusing us with too much data.

6 Conclusions

None of the systems described in this paper – KidPix, the SonicFinder, SoundShark and ARKola, or EAR – were explicitly designed for educational purposes. But all have something to say about how to make such systems easier to use, more motivating, more supportive of group work, and better able to guide students in their learning about complex, rich environments. Sounds can make systems easier to learn and more enjoyable to use. They are useful in supporting coordination without enforcing togetherness, because they can allow background information to be conveyed using the auditory channel while foreground, task-specific information can be provided visually. Finally, by providing only a subset of information, it is possible to use auditory icons to guide exploration along trajectories that are useful and desirable. So while the idea of designing educational software to be more noisy may have seemed counter-intuitive at the start of this paper, I hope that it is now possible for readers to imagine a day when teachers will be pleading with their students to make *more* noise.

References

Buxton, W., and Moran, T. (1990). EuroPARC's integrated interactive intermedia facility (iiif): Early experiences. In *Proceedings of the IFIP WG8.4 Conference on Multi-User Interfaces and Applications* (Herakleion, Crete, September 1990).

Cohen, J. (1993). Kirk here: Using genre sounds to monitor background activity. *Adjunct Proceedings of INTERCHI'93* (Amsterdam, May 1993). ACM, New York.

Gaver, W. W. (1989). The SonicFinder: An interface that uses auditory icons. Human-Computer Interaction. 4 (1).

Gaver, W. W. (1991). Sound support for collaboration. *Proceedings of the Second European Conference on Computer-Supported Collaborative Work* (Amsterdam, September 1991) Kluwer, Dordrecht.

Gaver, W. W. (1993a). How do we hear in the world? Explorations of ecological acoustics. *Ecological Psychology* (5) 4.

Gaver, W. W. (1993b). What in the world do we hear? An ecological approach to auditory source perception. *Ecological Psychology* (5) 1.

Gaver, W. W., & Smith, R. B. (1990). Auditory icons in large-scale collaborative environments. *Proceedings of Human-Computer Interaction – Interact'90* (Cambridge, U.K., August 1990) North Holland, Amsterdam.

Gaver, W. W., Moran, T., MacLean, A., Lövstrand, L., Dourish, P., Carter, K., & Buxton, W. (1992). Realizing a video environment: EuroPARC's RAVE system. *Proceedings of CHI'92* (Monterey, CA., May 1992). ACM, New York.

Gaver, W. W., Smith, R. B., & O'Shea, T. (1991). Effective sounds in complex systems: The ARKola simulation. *Proceedings of CHI 1991,* New Orleans, April 28 - May 2, 1991, ACM, New York.

Hutchins, E. , Hollan, J. , & Norman, D. (1986). Direct manipulation interfaces. In D. A. Norman & S. W. Draper (Eds.), *User centered system design: New perspectives on human-computer interaction.*. Hillsdale, NJ: Lawrence Erlbaum .

Laurel, B., Oren, T., and Don, A. (1990). Issues in multimedia interface design: Media integration and interface agents. *Proceedings of CHI'90* (Seattle, April 1990) ACM: New York.

Lövstrand, L. (1991). Being selectively aware with the Khronika system. In *Proceedings of ECSCW'91* (Amsterdam, September 1991).

Smith, R. (1987). The Alternate Reality Kit: an example of the tension between literalism and magic. *Proceedings of CHI + GI 1987* (Toronto, April 1987) ACM, New York.

Smith, R. (1989). A prototype futuristic technology for distance education. *Proceedings of the NATO Advanced Workshop on New Directions in Educational Technology.* (Nov. 1988, Cranfield, UK.)

Prosody in experimental dialogues

Ronald Geluykens and Marc Swerts

Institute for Perception Research (IPO),
P.O. Box 513, Nl-5600 MB Eindhoven, The Netherlands
E-mail: geluyken@heiipo5.bitnet; Fax: +32-3-220 44 20

It is widely recognized that prosody plays a predominant role in the organization of discourse. More specifically, it may be used both to signal information structure [topic boundaries] and to govern interaction [turn boundaries]. From the recent discourse-analytic literature dealing with prosody in conversation, it is not clear, however, how these two dimensions may interact. Moreover, prosodic analyses have mainly been limited to the study of final pitch movements only, though one often has the impression that discourse units can also have more global prosodic correlates. Both these questions, i.e. (i) interaction of prosodic cues to topics and turns and (ii) local and global prosodic demarcation of discourse units, are addressed in this paper in an experimental manner.

From five pairs of test subjects, three types of discourse were elicited, each consisting of a speaking and a listening task. In all conditions, speakers were not allowed to use lexical or syntactic cues to clarify discourse structure, and therefore could only rely on prosodic devices. In condition 1, a monologue setting designed to test information structure independently from turn-taking considerations, speakers were instructed to describe series of concatenations of differently colored geometrical figures [e.g.'a red triangle'] in such a way that major breaks between series became apparent, while listeners had to try and detect these breaks and mark them on an answer sheet. Condition 2 involved enforced turn-taking independently from information structure: participants were asked to take over turns at predetermined points in their descriptions of a simple row of geometrical figures. In condition 3, finally, both variables --topic and turn boundaries-- were combined to create a more complex setting, as topical breaks could occur within turns.

The use of prosody as a marker of discourse structure appears to depend largely on the type of discourse setting. Firstly, on a local level, the nature of final pitch movements is different across conditions: it is not the case that finality is always signalled by means of a falling contour. Secondly, on a more global level, several prosodic devices --accent shifts from noun to adjective, topline declination, and relative heights of pitch maxima within NPs-- are employed to signal discourse structure. These tendencies increase as a function of the complexity of the speaking task; in other words, they are most outspoken in condition 3.

This indicates that speakers indeed employ prosody both on a local and a global level for structuring dialogue discourse, both from the point of view of topic organization and from the point of view of interaction. These acoustic results will be discussed at the workshop, together with their perceptual evaluation.

The Writing Instruction Script Hebrew (WISH) System

Jon W. McKeeby[1], Yael M. Moses and Rachelle S. Heller[1]
[1] Department of Electrical Engineering and Computer Science
[2]Department of Classical Languages
George Washington University, Washington, D. C. 20052, USA

Abstract
This paper will describe an effective tool for the teaching of handwriting of languages that utilize non-Roman character sets being developed at George Washington University. The tool utilizes a microcomputer, videodisc player, and a stylus and a graphics tablet, to enable students to review the handwriting process, test handwriting abilities, while instructing the student on handwriting skills.

Keywords: Multimedia, Computer Assisted Instruction, Foreign Language

1.0 Background

In the early seventies two self-proclaimed successful CAI handwriting systems for non-Roman languages were developed. Abboud's system (1972), designed to instruct students in the writing of Arabic, consisted of a computer and a grease pencil. Students were instructed to write letters on the computer screen by using the grease pencil. After the student had completed this task, the student was presented the correct letter formation on the screen requiring the students to analyze their own handwriting skills.

Through an empirical study, Abboud reported that students taught using four to eight hours of CAI instruction and four hours of classroom instruction performed significantly better than students taught through a programmed instruction course or those students taught through the audio-lingual method. However, it should be noted that the audio-lingual methods base its instruction on listening and speaking and very little or no writing. In addition to the increased proficiency in handwriting mechanics, students that used the CAI system had a better attitude about class work and were more interested in continuing their Arabic language education.

Two problems of this system include the unconventional handwriting positioning and the inability to provide feedback relating to the students inputted letter. First, the normal handwriting position consists of the student resting his/her arm on a desktop surface, as opposed to writing a letter on a computer screen with his/her hand positioned freely in air. The second problem results from the students evaluating their own written letter. Even though the students are able to gain some visual feedback from examining a written letter against examples of the actual letter, their evaluations may be interpreted incorrectly. In addition, the comparison of a written letter and a target letter does not provide the student with information about the letter formation mechanics. Thus, the student's evaluation of his/her performance is, at best, incomplete.

The system by Chuang and Chen (1973) was designed to teach the writing techniques, assist in the handwriting process stroke by stroke, evaluate student's writing, and to provide informational feedback to aid the student correcting mistakes in letter formation and letter appearance. The developed system allowed students

to input letters by using a pen-stylus and a graphics tablet. Students were then provided visual feedback by displaying the inputted letter and the actual letter as well as instructional feedback describing the student's handwriting mechanics used to write a letter. Even though a completed system was described, no usability and empirical results were not provided and the educational effects are not known.

1.1 Difficulties in Creating Successful Handwriting Instruction System

The major reasons for the limited success of the development of CAI handwriting systems include the extensive time necessary to create effective systems, input and output technologies (i.e., graphic tablets, graphical computer displays, video output), the complex problem of recognizing written letters, and the inability to present a clear and complete instructional system.

In order to present students with complete feedback pertaining to their handwriting performance, it is necessary to provide realistic input techniques and visual feedback. Without the use of a handwriting or graphics tablet, the ability to enter written letter data would be impossible.

In order to present the student with complete analysis about an inputted letter, the inputted letter must be compared against a target letter. An instructional writing system is not true letter recognition in the sense that the letter that is being compared to the inputted letter is known, thus it is not necessary to sort through a complete set of letters. The written letter recognition, the comparison of an inputted letter and a target letter, is complex because of the variability of writing style (i.e., proportion size and shape appearance) from one individual to another (Stallings, 1975).

Even though a picture and a description of the writing process for a letter may provide an adequate description of the handwriting process, animation of the letter development process and a live demonstration conducted by the instructor allows the student to visualize the complete handwriting process for each letter of the new alphabet. Thus, a demonstration of the handwriting process is essential (Smith, 1987) in order to effectively instruct students.

2.0 The WISH System

The long term objective for the WISH (Writing Instruction Script in Hebrew) project is to provide an instructional non-Roman language handwriting system for Hebrew, Arabic, and other non-Roman languages. However, because of the simplicity of the cursive script of Hebrew (characters are not connected as in English cursive script and it is noncontextual as in Arabic) and the interest of a Hebrew professor to aid in the development of the content area, the initial attempt focused on a CAI system to teach the handwriting of the cursive script of Hebrew.

WISH will teach the correct formation of each of the twenty-seven cursive letters of the Hebrew language. The requirements of WISH are to allow students to observe the writing process, guide the student in writing each character, examine the characters being drawn and completed, and provide immediate instructional feedback to aid the student in correcting errors in character formation and appearance.

WISH handwriting development consists of two phases: a presentation stage and an interactive writing stage. The presentation stage uses video sequences and animation to guide students to form the cursive letters of the Hebrew alphabet. These segments allow students to observe the actual positioning of the instructor's pen, hand, arm, and body as well as the character formation and appearance on paper. These video segments reinforce the handwriting process and are accessible for the student as reference at anytime. Animation sequences are used to focus on the sequential formation of each character, stressing the character formulation and appearance attributes for each character. Descriptions of the important attributes of the letter are provided. Students can trace the cursive letter with the stylus and the Kurta tablet.

The presentation stage guides the student in the correct formation of Hebrew cursive letters. However, to teach the handwriting process and formation of each letter, it is necessary to evaluate a student's handwriting abilities and provide feedback to improve his/her handwriting. After the presentation stage for a given Hebrew cursive letter, the student enters the letter writing section of the interactive writing stage. Students use a stylus on a Kurta tablet to write many instances of the cursive letter corresponding to the square-script letter that is displayed on the computer screen (Figure 1). During the interactive writing stage, students' handwriting abilities are examined and evaluated both by student self-inspection and by computer generated evaluations.

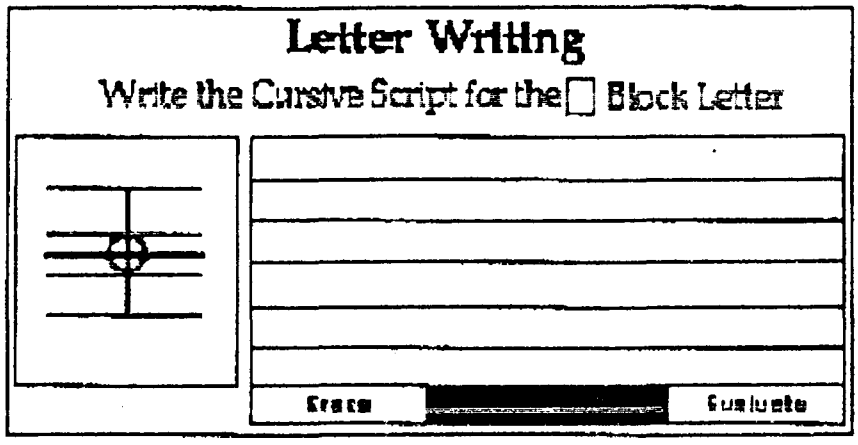


Figure 1. The Letter Writing Screen

As the student writes a letter, the written letter's (x,y) points and its boundary box are calculated. From the boundary box, the letter is resized with respect to the target letter and then overlaid on top of the target letter through a transformation of the (x,y) points. The visual evaluation provided by overlaying the written letter on the target letter provides immediate and constructive feedback to the student.

In addition to this immediate visual feedback, the correctness of the written letter is determined and feedback given to the user in order to improve his/her handwriting skills. To determine the correctness of the written letter, it is necessary to perform a detailed evaluation of the written letter by examining its attributes (Table 1).

Shape	:	Letter form/appearance
Proportional Size	:	The size in relation to the other written letters.
Position	:	The horizontal and vertical position of the whole letter as well as components of the letter.
Stroke Direction	:	Direction of writing stroke.
Stroke Order	:	The order in which strokes are written. Since most Hebrew cursive letters are one stroke letters, stroke order will also be used to refer to the connectivity of a one-stroke letter.
Proportional Spacing	:	The spacing between letters and lines of letters.

Table 1. Letter Attributes

By clicking on the desired letter in the letter writing section, the user is able to review the attribute evaluation of the selected letter (Figure 2). After reviewing the letter evaluation, the student is able to return to the writing of the letter. The student can continue to the next letter after successfully writing at least 5 letters of the current letter.

In addition to the letter writing section of the interactive writing stage, there is a practice section. The practice section allows the student to write the cursive letters within words to practice his/her writing skills. The only constraint on the practice section is that students are only allowed to practice letters completed in the letter writing stage. Here, too, the written target letter is overlaid on the target letter and the student is able to review the attribute evaluation of the written target letter by selecting the desired written letter.

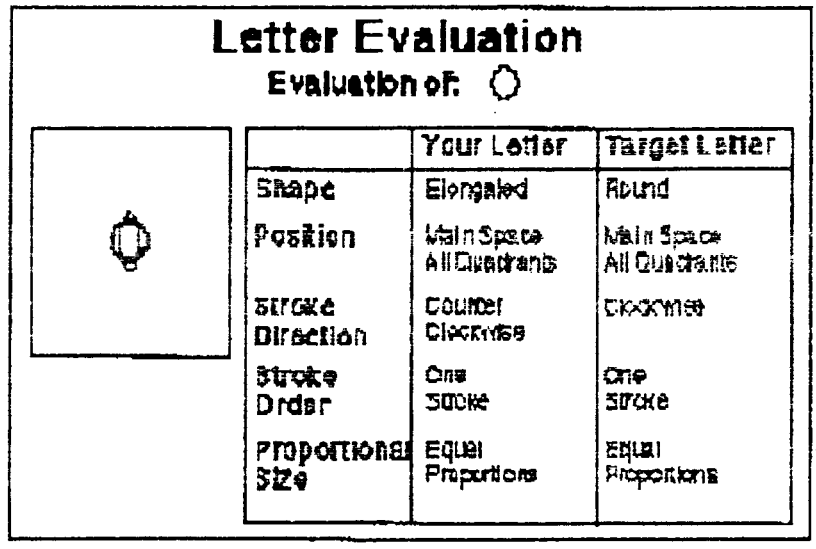


Figure 2. The Letter Evaluation Screen

2.1 The Attribute Evaluation Module

Since the target letter is known, a true letter recognition algorithm is not needed, but rather a technique to match the written letter to the target letter. For example, if the computer system instructs the student to write the letter "A" and the student writes the letter "B", the algorithm will determine that the written letter was a bad letter "A" and then describe why it was a bad letter "A", never identifying it as a "B".

In addition, to provide constructive feedback to the student, the evaluation algorithm must be able to identify the reasons the written letter was a "good" or "bad" match. To do so, it is necessary to perform a detailed evaluation of the written letter by examining each attribute of the written letter. As the algorithm evaluates each attribute, a list of feedback responses is created. The list describes the strengths and weaknesses of the written letter with respect to each letter attribute for the target letter. In addition, a guide is provided to the student to help increase the correctness in letter formation and appearance .

2.1.1 The Hebrew Cursive Alphabet

In order to identify the attributes of the Hebrew cursive alphabet, it was necessary to determine a set of general characteristics for Hebrew cursive letters as well as the letter attributes for each character of the Hebrew alphabet. A detailed attribute listing for each letter and identified the optimal letter as well as acceptable matches was created.

The shape attribute of the written letter is the actual points of the letter entered through the use of the stylus and the Kurta tablet seen by the user as well as the collection of entered points. The other attributes of the written cursive letter are determined through the written letter's characteristics: bounding area, center, starting point, ending point, and the order of strokes, and the character points. From these characteristics alone, the stroke direction and the stroke order of the written letter are determined and evaluated.

In order to determine a letter's proportional, proportional spacing, and position attributes, the written letter's characteristics are used with a space and quadrant system (Figure 3). The space component divides a letter into three vertical spacing areas, the lower space, the main space, and the upper space (all three spaces are of equal distance). The quadrant system is used to identify the center and horizontal positioning of a letter. The positioning of the quadrant and the distance of the vertical spacing are used to determine a letter's proportional attribute while the center of the quadrant system and the three spaces are used to determine a characters position.

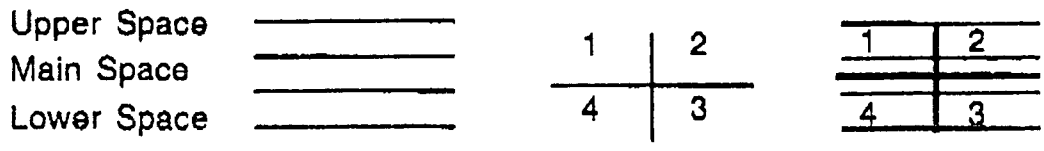


Figure 3. The Space, Quadrant, and Space and Quadrant System for the Hebrew Letters.

2.1.2 Shape Evaluation

The most difficult of all letter attributes to evaluate is the letter's shape. For the WISH system, the visualization of the character as well as a simple template matching algorithm are used to evaluate the correctness of the letter. Given that many of the letters of the Hebrew cursive alphabet are comprised of arcs and straight lines it was possible to develop a mathematical equation to represent many letters of the Hebrew cursive letter alphabet. From the size and proportion values of the written letter and the mathematical equation of the target letter, a template is created. For those letters that do not lend themselves to an equation, the digitized form of this letter is used as the template. The written letter is compared with the template. By using the size and proportion values of the written letter, the created template is equivalent in size to the written letter. The creation of the comparison template in real-time reduces the complications of the template matching that are inherent to matching written characters.

The comparison process consists of counting the number of black and white points in the sample that match the defined points in the template. If the written letter and the template are greater than 85% equivalent, then the shape of the entered character is correct. The written letter is compared to the created template because the written character has gaps between points because of the sampling rate and resolution of the tablet.

2.1.3 Proportional Size Evaluation

During the introduction of the WISH system, the student is provided a letter sample screen and is instructed to write samples of the *Samex* letter. From these, the student's proportional size value is calculated. The size of the character is determined by forming a bounding box of the character and calculating the length and width of the bounding box. The size of the character is used to evaluate the proportional size attribute of the character by comparing the size of the entered character to the student's proportional size value.

2.1.4 Position Evaluation

In addition to the proportional size value, the written sample letters are used to develop a template boundary box. This template boundary box involves the quadrant and component space system used to describe Hebrew cursive letters. The template boundary box can then be centered on top of the written letter and used to determine the correctness of the positioning attribute of the written letter.

2.1.5 Stroke Direction Evaluation

Stroke direction is evaluated throughout the writing of the letter in reference to the stroke direction and position of the target letter attribute. For each target letter, a listing of the changes of directions are maintained. As the stroke direction of the written letter changes during the writing process, its stroke direction list is updated. The stroke direction of the written letter is determined through the use of the previous and current (x,y) coordinates. Upon completion of the letter, the stroke direction list of the written letter is compared to the target listing with all differences noted in order to provide student feedback.

2.1.6 Stroke Order Evaluation

The stroke order of the entered letter is determined by ordering the strokes as they are written by the student and comparing them to the order of strokes of the expected letter. Since most letters only use one stroke, stroke order is also used to refer to connectivity. For letters that are only one stroke, the character is considered to be disconnected if the student lifts the stylus beyond the height of the proximity of the pen stylus and graphics tablet while writing.

2.1.7 Proportional Spacing Evaluation

Proportional spacing, the space between letters and lines, is only calculated when the student writes sequences of letters or words during the practice section. The distances are calculated by calculating the left, right, upper, and lower spacing between characters. These calculations are performed by subtracting components of the bounding boxes.

3.0 Conclusion

An innovative and effective interactive learning environment for students to learn the writing mechanics needed to produce characters of a non-Roman language has been described. Currently, the primary objective is to create a CAI system to effectively teach the handwriting of Hebrew cursive script characters. Usability testing and educational impact studies are planned for the 1993-94 academic year. Modifications to the Writing Instruction Script Hebrew (WISH) system will be made to enable it to effectively teach other non-Roman languages, such as Ancient Greek, Russian, Chinese, Japanese, Korean, and Arabic, as well.

References

Abboud, V.C. (1972, March). The computer as an instructional device for the Arabic writing system. Computers and the Humanities, 6, 4, 195-207.

Brod, R.I. (1988), Foreign Language Enrollments in US Institutions of Higher Education. ADFL Bulletin 1(19), 39-44.

Brown, R.M. (December, 1965). On-line computer recognition of handprinted characters. IEEE Transactions on Electronic Computers, 750-752.

Casey, R. & Nagy, G. (1966, February). Recognition of printed Chinese characters. IEEE Transactions on Electronic Computers, EC-15, 1, 91-101.

Chuang, H.Y., Chen, S. (1973, August). "Computer Aided Instruction in Chinese Characters". Proceedings of the first International symposium on computers and Chinese input/output systems, 599 - 616.

Govindan, V.K. & Shivaprasad, A.P. (1990). Character recognition - A review. Pattern Recognition, 23, 7, 671-683.

Heller, R.S. & Moses, Y.M (July 1990). OT L'OT: A pilot program - An interactive Computer aided instruction tutorial for the writing of the Hebrew alphabet. A Proposal to IBM Corporation. George Washington University, Washington, D. C.

Smith, W. (1989). Modern Technology in Foreign Language Education: Applications and Projects. National Textbook Company. Lincolnwood, Illinois.

Stallings, W.W. (1975). Approaches to Chinese character recognition. Pattern Recognition, 8, 87-98.

Stallings, W.W. (1977). Chinese character recognition. In K.S. Fu (Ed.), Communication Syntactic Pattern Recognition: Applications & Cybernetics 14. (pp. 95-123). Berlin, Germany: Springer-Verlag.

Ward, J.R. & Blesser, B. (September 1985). Interactive recognition of handprinted characters for computer input. IEEE Computer Grapics and Applications. 24 - 37.
Woolsey, Kristina Hooper. (Spring 1991). A discussion with Jon McKeeby, Racheile Heller, and Kristina Hooper Woolsey regarding the WISH system.

The Learner's Partner: Foreign Language Learning and Real World Encounters

Stacie L. Hibino

Computer Science and Engineering, University of Michigan, 1101 Beal Avenue, Ann Arbor, MI 48109-2110, USA

The Learner's Partner is a pedagogical model using interactive multimedia for foreign language learning. This model is designed for individual student use (or for use by pairs of students) and focuses on several aspects of language acquisition, including: viewing and listening, reading, writing, speaking, comprehension, and cultural understanding. A variety of activities are provided at increasing levels of interactivity and difficulty, thus providing some scaffolding to the student while addressing individual learning styles and needs.

Video is at the heart of the model, providing cultural context, modeling, and real world encounters. Two types of video segments are used — real-life scenarios (e.g., buying a bus ticket) and short cultural reports. Activities for the scenarios focus primarily on comprehension and speaking skills, leading up to role playing. Activities for the cultural reports focus on comprehension and writing, leading up to student research reports.

Applications using the Learner's Partner (LP) model have been developed in French, Spanish, and Chinese, for first and second year students. LP activities in Hebrew are also currently being developed. The applications have not yet been incorporated into classroom use, but some preliminary pilot testing has been done.

My role in the LP has not only involved design, but also the development of a template. The LP template incorporates authoring tools for each of the activities, thus allowing foreign language experts (i.e., end-user programmers) to easily develop LP applications while still having some room for creativity.

This poster session will display and describe each of the LP activities, as well as some of the authoring tools used by the developers.

Acknowledgements

Learner's Partner applications are being designed and developed by several members of Project FLAME (Foreign Language Applications in a Multimedia Environment) at the University of Michigan. Besides myself, the design and development team include: Professor Edna Coffin (project director; Hebrew), Gonzalo Silverio (Spanish), Joanna Porvin (French), and Pamela Colquitt (Chinese).

New Human-Computer Interaction Techniques

Robert J.K. Jacob
Human-Computer Interaction Lab, Naval Research Laboratory, Washington, D.C., U.S.A.

Abstract. This chapter describes the area of human-computer interaction technique research in general and then describes research in several new types of interaction techniques under way at the Human-Computer Interaction Laboratory of the U.S. Naval Research Laboratory.

1. Introduction

Tufte [12] has described human-computer interaction as two powerful information processors (human and computer) attempting to communicate with each other via a narrow-bandwidth, highly constrained interface. A fundamental goal of research in human-computer interaction is, therefore, to increase the useful bandwidth across that interface. A significant bottleneck in the effectiveness of educational systems as well as other interactive systems is this communication path between the user and the computer. Since the user side of this path is difficult to modify, it is the computer side that provides fertile ground for research in human-computer interaction. This chapter describes interaction technique research in general and then describes research in several new types of interaction techniques under way at the Human-Computer Interaction Laboratory of the U.S. Naval Research Laboratory.

Interaction techniques provide a useful focus for human-computer interaction research because they are specific, yet not bound to a single application. An interaction technique is a way of using a physical input/output device to perform a generic task in a human-computer dialogue [1]. It represents an abstraction of some common class of interactive task, for example, choosing one of several objects shown on a display screen. Research in this area studies the primitive elements of human-computer dialogues, which apply across a wide variety of individual applications. The basic approach is to study new modes of communication that could be used for human-computer communication and develop devices and techniques to use such modes. The goal is to add new, high-bandwidth methods to the available store of input/output devices, interaction techniques, and generic dialogue components. Ideally, research in interaction techniques starts with studies of the characteristics of human communication channels and skills and then works toward developing devices and techniques that communicate effectively to and from those channels. Often, though, the hardware developments come first, people simply attempt to build "whatever can be built," and then HCI researchers try to find uses for the resulting artifacts.

2. Eye Movement-Based Interaction Techniques

One of the principal thrusts in interaction technique research at NRL has been eye movements [5, 6, 9]. We have been interested in developing interaction techniques based on eye movements as an input from user to computer. That is, the computer will identify the point on its display screen at which the user is looking and use that information as a part of its dialogue with the user. For example, if a display showed several icons, a user might request additional information about one of them. Instead of requiring the user to indicate which icon was desired by pointing at it with a mouse or by entering its name with a keyboard, the computer can determine which icon the user was looking at and give the information on it immediately.

Our approach to this interaction medium is to try to make use of natural eye movements. This work begins by studying the characteristics of natural eye movements and then attempts to recognize appropriate patterns in the raw data obtainable from an oculometer, turn them into tokens with higher-level meaning, and design interaction techniques for them around the known characteristics of eye movements. A user interface based on eye movement inputs has the potential for faster and more effortless interaction than current interfaces, because people can move their eyes extremely rapidly and with little conscious effort. A simple thought experiment suggests the speed advantage: Before you operate any mechanical pointing device, you usually look at the destination to which you wish to move. Thus the eye movement is available as an indication of your goal before you could actuate any other input device.

However, people are not accustomed to operating devices in the world simply by moving their eyes. Our experience is that, at first, it is empowering to be able simply to look at what you want and have it happen, rather than having to look at it and then point and click it with the mouse. Before long, though, it becomes like the Midas Touch. Everywhere you look, another command is activated; you cannot look anywhere without issuing a command. The challenge in building a useful eye movement interface is to avoid this Midas Touch problem. Carefully designed new interaction techniques are thus necessary to ensure that they are not only fast but that use eye input in a natural and unobtrusive way. Our approach is to try to think of eye position more as a piece of information available to a user-computer dialogue involving a variety of input devices than as the intentional actuation of the principal input device.

A further problem arises because people do not normally move their eyes in the same slow and deliberate way they operate conventional computer input devices. Eyes continually dart from point to point, in rapid and sudden "saccades." Even when a user thinks he or she is viewing a single object, the eyes do not remain still for long. It would therefore be inappropriate simply to plug in an eye tracker as a direct replacement for a mouse. Wherever possible, we therefore attempt to obtain information from the natural movements of the user's eye while viewing the display, rather than requiring the user to make specific trained eye movements to actuate the system.

We partition the problem of using eye movement data into two stages. First we process the raw data from the eye tracker in order to filter noise, recognize fixations, compensate for local calibration errors, and generally try to reconstruct the user's more conscious intentions from the available information. This processing stage uses a model of eye motions (fixations separated by saccades) to drive a fixation recognition algorithm that converts the continuous, somewhat noisy stream of raw eye position reports into discrete tokens that represent user's intentional fixations. The tokens are passed to our user interface management system, along with tokens generated by other input devices being used simultaneously, such as the keyboard or mouse.

Next, we design generic interaction techniques based on these tokens as inputs. The first interaction technique we have developed is for object selection. The task is to select one object from among several displayed on the screen, for example, one of several file icons on a desktop. With a mouse, this is usually done by pointing at the object and then pressing a button. With the eye tracker, there is no natural counterpart of the button press. We reject using a blink for a signal because it detracts from the naturalness possible with an eye movement-based dialogue by requiring the user to think about when he or she blinks. We tested two alternatives. In one, the user looks at the desired object then presses a button on a keypad to indicate his or her choice. The second alternative uses dwell time—if the user continues to look at the object for a sufficiently long time, it is selected without further operations.

At first this seemed like a good combination. In practice, however, the dwell time approach proved much more convenient. While a long dwell time might be used to ensure that an inadvertent selection will not be made by simply "looking around" on the display, this mitigates the

speed advantage of using eye movements for input and also reduces the responsiveness of the interface. To reduce dwell time, we make a further distinction. If the result of selecting the wrong object can be undone trivially (selection of a wrong object followed by a selection of the right object causes no adverse effect—the second selection instantaneously overrides the first), then a very short dwell time can be used. For example, if selecting an object causes a display of information about that object to appear and the information display can be changed instantaneously, then the effect of selecting wrong objects is immediately undone as long as the user eventually reaches the right one. This approach, using a 150-250 ms. dwell time gives excellent results. The lag between eye movement and system response (required to reach the dwell time) is hardly detectable to the user, yet long enough to accumulate sufficient data for our fixation recognition and processing. The subjective feeling is of a highly responsive system, almost as though the system is executing the user's intentions before he or she expresses them. For situations where selecting an object is more difficult to undo, button confirmation is used rather than a longer dwell time.

Other interaction techniques we have developed and are studying in our laboratory include: continuous display of attributes of eye-selected object (instead of explicit user commands to request display); moving object by eye selection, then press button down, "drag" object by moving eye, release button to stop dragging; moving object by eye selection, then drag with mouse; pull-down menu commands using dwell time to select or look away to cancel menu, plus optional accelerator button; forward and backward eye-controlled text scrolling.

Eye movement-based interaction techniques exemplify an emerging new style of interaction, called non-command-based [10]. Previous interaction styles all await, receive, and respond to explicit commands from the user to the computer. In the non-command style, the computer passively monitors the user and responds as appropriate, rather than waiting for the user to issue specific commands. Because the inputs in this style of interface are often non-intentional, they must be interpreted carefully to avoid annoying the user with unwanted responses to inadvertent actions. Our research with eye movements provides an example of how these problems can be attacked.

3. Three-Dimensional Interaction

Another area of interaction technique research at NRL has been an investigation of three degree of freedom input [7]. In studying interaction techniques, each new piece of hardware that appears raises the question "What tasks is this device good for, and how should it be incorporated into interface designs?" Such questions are typically answered specifically for each new device, based on the intuition and judgment of designers and, perhaps, on empirical studies of that device. Our work in three degree-of-freedom input provides an example of how greater leverage can be achieved by answering such questions by reasoning from a more general predictive theoretical framework, rather than in an *ad hoc* way.

We begin by posing the question for the three-dimensional position tracker, such as the Polhemus 3SPACE or Ascension Bird trackers. While directly answering the question "What is a three-dimensional tracker good for?" we also try to shed light on the next level question, i.e., "How should you answer questions like 'What is a three-dimensional tracker good for?'" Concepts such as the logical input device provide descriptive models for understanding input devices, but they tend to ignore the crucial pragmatic aspects of haptic input by treating devices that output the same information as equivalent, despite the different subjective qualities they present to the user. Taxonomies and other frameworks for understanding input devices have tended to hide these pragmatic qualities or else relegate them to a "miscellaneous" category, without further structure.

Instead, we draw on the theory of processing of perceptual structure in multidimensional space [2,3]. The attributes of objects in multidimensional spaces can have different dominant perceptual structures. The nature of that structure, that is, the way in which the dimensions of the space combine perceptually, affects how an observer perceives an object. We posit that this distinction between perceptual structures provides a key to understanding performance of multidimensional input devices on multidimensional tasks. Hence two three-dimensional tasks may seem equivalent, but if they involve different types of perceptual spaces, they should be assigned to correspondingly different input devices.

The three-dimensional position tracker can be viewed as a three-dimensional absolute-position mouse or data tablet; it provides continuous reports of its position in three-space relative to a user-defined origin. The device thus allows a user to input three coordinates or data values simultaneously and to input changes that cut across all three coordinate axes in a single operation. (A mouse or trackball allows this in only two dimensions.) Such a device is obviously useful for pointing in three-space, but it is also applicable in many other situations that involve changing three values simultaneously. We considered two tasks that both involve three degrees of freedom, i.e., that require adjusting three variables. For comparison with the three-dimensional tracker, we used a conventional mouse (for two of the three variables in the tasks) and then provided a mode change button to turn the mouse temporarily into a one-dimensional slider for the third variable.

A naive view of these two alternatives suggests that the three-dimensional tracker is a superset of the two-dimensional mouse, since it provides the same two outputs plus a third. Thus the three-dimensional tracker should always be used in place of a mouse (assuming ideal devices with equal cost and equal accuracy), since it is always at least as good and sometimes better. Our intuition tells us that this is unlikely—but why? The goal of this research is to develop a firmer foundation from which to draw such judgments. To do this, we extend Garner's theory of processing of perceptual structure [3], first developed with fixed images, to interactive graphical manipulation tasks and thereby use it to shed light on the selection of multidimensional input devices. Garner observed that relationships between the attributes of an object can be perceived in two ways that differ in how well the component attributes remain identifiable. Some attributes are *integrally* related to one another—the values of these attributes combine to form a single composite perception in the observer's mind, and each object is seen as a unitary whole; while other attributes are *separably* related—the attributes remain distinct, and the observer does not integrate them, but sees an object as a collection of attributes.

Our hypothesis is that the structure of the perceptual space of an interaction task should mirror that of the control space of its input device. To examine it, we considered two interactive tasks, one set within an integral space and one in a separable one, and two input devices, one with integral dimensions and one, separable. This yields a two by two experiment, with four conditions. We expect performance on each task to be superior in the condition where the device matches that task in integrality/separability. That is, the interaction effect between choice of task and choice of device should far exceed the main effects of task or device alone.

For the integral three-attribute task in the experiment, the user manipulates the x-y location and the size of an object to match a target, since location and size tend to be perceived as integral attributes; for the separable task, the user manipulates the x-y location and color (lightness or darkness of greyscale) of an object to match a target, since location and color are perceived separably. The difference in perceptual structure between these two tasks is in the relationship of the third dimension (size or greyscale) to the first two (x and y location); in all cases, the x and y attributes are integral.

For the integral device condition, we use a Polhemus tracker, which permits input of three integral values. For the separable condition, we use a conventional mouse, which permits two

integral values, to which we added a mode change to enable input of a third—separable—value. Our hypothesis predicts that the three degree of freedom input device will be superior to the two degree of freedom (plus mode change) device only when the task involves three integral values, rather than in all cases, as with the naive hypothesis mentioned above.

Our experimental results strongly supported this hypothesis. We found that neither device is uniformly superior to the other in performance. Instead, we find significantly better performance in the experimental conditions where the task and device are both integral or both separable and inferior performance in the other two conditions. These results support our extension of the theory of perceptual space to interaction techniques, which predicts that the integral task (size) will be performed better with the integral device (Polhemus) and that the separable task (greyscale) will be performed better with the separable device (mouse).

How might these results be used in designing controls for zooming and panning of a geographic display? Zooming and panning, taken together, involve three degrees of freedom. The most common design uses a mouse or trackball for two-dimensional panning and a separate control for zooming. We claim that a user typically does not really think of zooming or panning operations separably, but thinks rather of integral operations like "focus in on *that* area over there." The space is thus Euclidean, like that of the size task in the experiment, and, therefore, making the user do the two separately violates perceptual compatibility. It would be more natural to permit a user to make a gesture that performs the overall operation he or she had in mind, using an integral three-dimensional input device. The user moves the puck around in a volume directly in front of the display screen. Moving it in the x or y direction parallel to the display surface causes panning; moving it perpendicular to the display (directly toward or away from it) causes zooming. The user typically moves the puck in all three dimensions simultaneously, resulting in some combination of zooming and panning and directly reaches the view of interest. We have demonstrated a mockup of this application.

4. Egocentric Projection

Recent work at NRL has greatly extended the notion of a three degree of freedom panning and zooming device to the concept of egocentric projection (to be discussed further in final version of this chapter).

5. Dialogue Interaction Techniques

Another direction in our research is the notion of dialogue interaction techniques [8, 11]. In a direct manipulation or graphical interface, each command or brief transaction exists as a nearly independent utterance, unconnected to previous and future ones from the same user. Real human communication rarely consists of such individual, unconnected utterances, but rather each utterance can draw on previous ones for its meaning. It may do so implicitly, embodied in a conversational focus, state, or mode, or explicitly. Most research on the processes needed to conduct such dialogues has concentrated on natural language, but some of them can be applied to any human-computer dialogue conducted in any language. A direct manipulation dialogue is conducted in a rich graphical language using powerful and natural input and output modalities. The user's side of the dialogue may consist almost entirely of pointing, gesturing, and pressing buttons, and the computer's, of animated pictorial analogues of real-world objects. A dialogue in such a language could nevertheless exhibit useful dialogue properties, such as following focus. For example, a precise meaning can often be gleaned by combining imprecise actions in several modes, each of which would be ambiguous in isolation. We thus attempt to broaden the notion of interaction techniques in these two dimensions (multiple transactions and multiple modes).

A useful property of dialogue that can be applied to a graphical interface is focus [4]. The graphical user interface could keep a history of the user's current focus, tracking brief

digressions, meta-conversations, major topic shifts, and other changes in focus. Unlike a linguistic interface, the graphical interface would use inputs from a combination of graphical or manipulative modes to determine focus. Pointing and dragging of displayed objects, user gestures and gazes as well as the objects of explicit queries or commands all provide input to determine and track focus. Moreover, focus would not be maintained as a single object, but rather a history of the course of the user-computer dialogue. It is necessary to track excursions or digressions in the dialogue so focus can be restored as necessary. In addition, it is helpful to track focus by categories. This allows the user to refer to "the ship" even though the current focus is another object. In that case, the recent history of focus would be searched to find a ship of the appropriate category. Finally, focus is not necessarily a concrete object; it may be a class or category of objects ("all blue ships") or a more abstract entity ("the previous command").

Human dialogue often combines inputs from several modes. Deixis often involves a pointing gesture that does not precisely specify its object; the listener deduces the correct object from the context of the dialogue and, possibly, from integrating information from the hand gesture, the direction of the user's head, tone of his or her voice, and the like. A user could, similarly, give a command and point in a general direction to indicate its object. The interface would disambiguate the pointing gesture based on the recent history of its dialogue with the user and, possibly, by combining other information about the user from physical sensors. An imprecise pointing gesture in the general direction of a displayed region of a map could be combined with the knowledge that the user's recent commands within that region referred principally to one of three specific locations (say, river *R*, island *I*, and hill *H*) within the region and the knowledge that the user had previously been looking primarily at islands displayed all over the map. By combining these three imprecise inputs, the interface could narrow the choice down so that (in this example) island *I* is the most likely object of the user's new command.

We call these higher-level interaction elements dialogue interaction techniques, and we have begun designing and testing them in our laboratory. We are also developing a software architecture for handling these properties that span more than one transaction. It treats them as orthogonal to the usual lexical, syntactic, and semantic partitioning of user interface software. Our first system demonstrates the use of a focus stack in an interactive graphics editor. In the future, we will expand to a richer representation of dialogue than a stack, to support a wider range of dialogue interaction techniques.

6. Conclusions

This chapter has provided an overview of a variety of new human-computer interaction techniques we are studying and building at NRL. Interaction techniques like these, when applied to the design of specific interfaces, increase the useful bandwidth between user and computer. This seems to be the key bottleneck in improving the usefulness of all types of interactive computer systems, and particularly educational systems, which depend heavily on dialogues with their users.

Acknowledgments

I want to thank my colleagues at the Naval Research Laboratory, Dan McFarlane, Preston Mullen, Manuel Perez, Linda Sibert, and Jim Templeman. who have conducted much of the research I have described in this chapter. This work was sponsored by the Office of Naval Research.

References

1. J.D. Foley, A. van Dam, S.K. Feiner, and J.F. Hughes, *Computer Graphics: Principles and Practice*, Addison-Wesley, Reading, Mass. (1990).
2. W.R. Garner and G.L. Felfoldy, "Integrality of Stimulus Dimensions in Various Types of Information Processing," *Cognitive Psychology* **1** pp. 225-241 (1970).
3. W.R. Garner, *The Processing of Information and Structure*, Lawrence Erlbaum, Potomac, Md. (1974).
4. B.J. Grosz, "Discourse," pp. 229-284 in *Understanding Spoken Language*, ed. D.E. Walker, Elsevier North-Holland, New York (1978).
5. R.J.K. Jacob, "What You Look At is What You Get: Eye Movement-Based Interaction Techniques," *Proc. ACM CHI'90 Human Factors in Computing Systems Conference* pp. 11-18, Addison-Wesley/ACM Press (1990).
6. R.J.K. Jacob, "The Use of Eye Movements in Human-Computer Interaction Techniques: What You Look At is What You Get," *ACM Transactions on Information Systems* **9**(3) pp. 152-169 (April 1991).
7. R.J.K. Jacob and L.E. Sibert, "The Perceptual Structure of Multidimensional Input Device Selection," *Proc. ACM CHI'92 Human Factors in Computing Systems Conference* pp. 211-218, Addison-Wesley/ACM Press (1992).
8. R.J.K. Jacob, "Natural Dialogue in Modes Other Than Natural Language," in *Natural Dialogue and Interactive Student Modelling*, ed. R.-J. Beun, Springer-Verlag, Amsterdam (1993). (in press).
9. R.J.K. Jacob, "Eye Movement-Based Human-Computer Interaction Techniques: Toward Non-Command Interfaces," pp. 151-190 in *Advances in Human-Computer Interaction, Vol. 4*, ed. H.R. Hartson and D. Hix, Ablex Publishing Co., Norwood, N.J. (1993).
10. J. Nielsen, "Noncommand User Interfaces," *Comm. ACM* **36**(4) pp. 83-99 (April 1993).
11. M.A. Perez and J.L. Sibert, "Focus in Graphical User Interfaces," *Proc. ACM International Workshop on Intelligent User Interfaces*, Addison-Wesley/ACM Press, Orlando, Fla. (1993).
12. E.R. Tufte, "Visual Design of the User Interface," IBM Corporation, Armonk, N.Y. (1989).

Psychological Peculiarities of Man-Machine Communication in the Instructional Systems

Yelena Komissarova
Psychological Institute, Kiev, Ukraine

Human-computer communication in the instructional systems, compared to that in the ordinary/general purpose man-machine systems and to student-teacher interaction in the traditional instruction, has its own psychological peculiarities.

In terms of activity/task approach the goal of functioning of any ordinary man-machine system is solving of certain problem in particular domain of knowledge, and the dialogue interaction between man and computer is aimed at the fulfilment of this task. The main goal of instructional system is not to solve particular subject problem, but to create desirable changes in a student himself.

This feature causes further distinctions in man-machine communication in professional and instructional contexts which are as follows.

1. In the situation of instruction the man-machine interaction is a means for student's learning activity control. So, this interaction can be considered efficient if it allows to reach instructional objectives and goals.

2. In efficient instructional systems the subject of dialogue is not restricted to the subject content of the learning task solving activity. In this situation the communication between computer and student should influence the latter's cognitive activity immediately, facilitate its forming, promote self-consciousness and the rise of reflexion etc.

3. Besides external dialogue, instructional systems are (or should be) featured by student's internal dialogue; this point is crucial. Dialogue is regarded not as an exchange of remarks, but it arises due to existence of two positions, two views of the same problem. Promotion of internal dialogue provided by external man machine dialogue is indicative of efficiency of dialogue communication in instructional systems.

All the above-mentioned features ensue from each other and are characteristic for man-machine communication in the context of instruction. They should be taken into account while designing and evaluating instructional software. Designers should direct their efforts at and use all the technologies available for enhancing the internal dialogue of the student.

Comparing communicative aspect of instructional systems to live communication between student and teacher, one can elicit the following peculiarities.

1. Reverse asymmetry: in traditional instruction human communication is asymmetric in favour of teacher who is controlling the situation and the learning process. He usually initiates interaction with the student, takes decision about its course and interruptions, manages time, estimates student's activity and performance. However, in computer-assisted instruction the situation is quite different. Though keeping implicit control of the learning process as to instructional objectives and goals and ways of reaching them, instructional system should explicitly emphasize student's dominating position and initiative in starting or interrupting communication, in choosing his path through the curriculum.

2. Redefining of learning task: It is a widespread situation in CAI, when a student tends to change given learning task for a different one. One of the possible reasons for it can be emotional reaction that emerges in student due to judgemental attitude of instructional system. Student can accept estimate of his progress given by human teacher, but he usually wouldn't admit that evaluation of his performance by machine is legitimate. So he often tries to prove computer's stupidity by means of substituting the real learning task by similar but different one. In ordinary man-machine systems situations of that kind can't arise since computer is used as a tool for solving professional tasks. Designers of instructional software should be very careful in order to provide safe and comfortable atmosphere for students.

References

Машбиц, Е.И., Андриевская, В.Р., Комиссарова, Е.Ю.: Диалог в обучающей системе. Киев: Выща школа 1984.

Машбиц, Е.И.: Психологические основы управления учебной деятельностью. Киев: Выща школа 1987.

Some benefits and problems of adaptive action-oriented user-interfaces

Pertti Saariluoma Michael Miettinen

University of Helsinki
Department of Psychology
Fabianinkatu 28
00100 Helsinki 10

A presentation to be given in NATO ASI: Basics of Man-Machine Communication for the Design of Educational Systems
Eindhoven August 16.8.-24.8. 1993.

Abstract: Human actions are chunked. Low-level action components are organized into hierarchical systems and, normally, people do not think but rather use the highest level concepts. A user-interface which takes into account the hierarchical and chunked organization of human actions, we call action-oriented. In an action-oriented interface, all action components are hidden behind general action labels and when working with an action-oriented interface, the actions are called the action labels. A way of realizing action-oriented user-interface is to let the interface monitor human behavior and adaptively modify itself. In this paper, we discuss three experiments we conducted to study some psychological problems in the use of supervised adaptive interface which follows the action-oriented principles. In the experiments adaptive interface was shown to be a viable method of encouraging the use of action-oriented interface; computer-cuing in order to promote the subjects' use of action-oriented facilities.

1. Introduction

People normally chunk their actions into wholes (Chase and Simon 1973, Miller 1956, Ericsson and Kintsch 1991). For example, when taking something we do not think about moving shoulders and elbow joints, wrists and fingers but instead, we just take the object. Taking is an action and all the required subprocesses are subordinated under this one action label in our minds. Information processing, which is build on chunked actions, could be termed actionbased.

Human thinking and communication is action-based. The main goal is important and the operative details are as far as possible hidden behind the high-level action concepts. We believe that the chunked character of human actions should be respected in designing user-interfaces and we call a user-interface which is close to this ideal, action-oriented.
Instead of forcing users into manual and mentally loading process control, an action-oriented user-interface requires nothing but information about the user's intended actions. The name of the action, however complex, should be enough to get a computer to carry out the action with minimal, if any, human involvement.

Two conditions are required: Firstly, the concept of action oriented user-interface is polar. Interfaces are neither totally action-oriented nor standard, but more or less action-oriented. Secondly, an action-oriented interface is different from the notion of adaptive user-interface (Murray 1991, Wetzenstein, Ollenschläger and Wandke 1990). Action-oriented user interfaces need not be adaptive because they can be realized without any adaptation. Neither needs an adaptive user-interface to be action-oriented. However, adaptive interfaces provide interesting technical opportunities for the designing of action-oriented user-interfaces.

The technical accomplishment of the chunking required in action-oriented interfaces is easy in principle easy, because it can be realized by writing macros. In this way it is possible to improve action-orienting in most standard interfaces. However, the range of users' needs, the variation of the programs and the programming environments make the practical designing of action-oriented interfaces difficult. An ideal action-oriented interface presupposes knowledge about users' and user-groups' abilities, mental models, and intentions, and this information is, for the most part inaccessible in the planning and manufacture of human-computer interfaces.

The design of action-oriented interfaces can also be enhanced by the users who can modify the interfaces to make them more suitable for their abilities and personal needs. This process can be fostered by providing suitable software to aid the modification. Machine learning and adaptive interfaces offer promising perspectives for interactive modification (Murray 1991, Wetzenstein-Ollenschläger and Wandke 1990). The interfaces themselves may learn to register users' needs and abilities by monitoring their behavior. They may either modify themselves or at least provide easy-to-use methods for supervising the modification. Since computers cannot possibly know the users' intentions, this supervised modification seems to be preferable to unsupervised interface modification.

However, our main interests are not technical but psychological. We are attempting to understand the psychological processes involved in working with action-oriented interfaces. A priori, we assume that one of the most difficult challenges for psychology in developing action-oriented interfaces will be caused by learning. Individuals' work-patterns develop in the course of time and training, and hence their expectations concerning the user-interfaces change. The users should also learn to use the facilities supplied by the interface. They should know how to use their existing knowledge to make the interface best answer their needs.

2. First experiment

2.1 Introduction

Our main experimental task was menu navigation. This task was selected because it is known that novices prefer menus but, with increasing expertise may find them less convenient (Card 1984, Paap and Roske-Hofstrand 1988, Vandierendonck, van Hoe, de Soete 1988).
Consequently, the problems of learning and interface adaptiviness seemed to be more relevant in menu navigation than in, e.g., programming. Of course, this does not mean that we think menu navigation is the only task in which the idea of action-oriented interface is relevant. We consider the concept of action-oriented interface as a general and versatile concept which can be discussed in practically any interface context.

The user's knowledge is an essential factor of his behavior in a menu environment. This knowledge may be divided into task-knowledge and environmental knowledge. The former refers to the content specific knowledge about goals and contents of a current computational task and the latter, knowledge about the behavior of the interface. Some of users are accidental and do not have much information about the task or the interface while others are experts in both. To observe the effects of these factors in our experiment, the familiarity of the task-environment was varied.

A second factor which could affect users, is the task complexity. If tasks are simple, it would seem natural to assume that current practice would be sufficient. The benefits of the action-oriented interfaces should become greater when the task demands increase. This is why the task complexity was varied.

The main variable was macro construction. Human action in a menu environment, which does not allow the use of macros, was compared with an environment which allows macro operations. We predicted that the use of macros would greatly improve subjects' performance and, consequently, the navigation task times would decrease in the macro environment compared to the non-macro environment.

2.2 Method

2.2.1 Subjects

Thirty two subjects were used. They were computer-science students and cognitive science students at Helsinki University. All students had a good basic knowledge of computers and applications software. All were males except one. The age of the subjects ranged between 22-26 years. The subjects were divided into four equally-sized groups.

2.2.2 Materials

The experiments were run with a standard 386SX PC with VGA monitor. A menu navigation program called ACONA I (Action Oriented Navigation Program) with macro facility was written with Pascal. ACONA I is a basic menu environment. It presents a number of options for the user and the users' task in navigating is to select from various options by pressing the correspondingly numbered key. Navigation takes place by moving from one menu to another until the target is reached. When the target has been found, the system backtracks to the beginning and presents the next task. The task number and the number of tasks in the experiment are presented to the subjects in the middle of the first row of screen. The target is shown in the middle of the seventh row. The current menu system node is presented underlined in the middle of the tenth row and the menu items indicating the next menus below it are located under it. In this and the second experiment, the number of the alternatives in each menu was four. The maximum depth of the search tree was five and thus the total number of nodes was 161.

Macros are made by pressing the spacebar while in the initial menu, thus marking the first element of the macro when the target menu is found, the last element of the macro is defined by pressing the spacebar a second time . After the definition of the last element, the interface ask "Do you want to make a macro from the first element (e.g. Copenhagen) to the last element (e.g. Helsinki) (Y/N)?". If the answer is negative, no macro is made. If the answer is positive, the interface asks the subject name to the macro. The maximum number of

characters in a name was 8. Self naming was selected because the literature of command names supports the usie of mnemonic names and the literature of cue-validity has convincingly shown the superiority of self-naming (Barnard and Grudin 1988, Mäntylä 1986).

From this point on, the macro can be called up by pressing the esc-key. Then the program prints a macro index which lists the names of all the macros, numbered in alphabetical order. To call up a macro, the subject presses the correspondingly numbered key. The call up of a macro moves the cursor to the macro endpoint menu. In this way, macros allow the chunking of the menu navigation processes.

2.2.3 Procedure and design

In these experiments the subjects were told that they should act as salesmen or saleswomen. The subjects' task was to move from Copenhagen to another city or town as quickly as possible. Copenhagen was the root node in all subtasks and in all conditions. The menu system presented city names as options or menu items. Moving from one city or town to naturally required normal menu navigation.

The depth of the search was varied by presenting target cities with either the depth of two or four. The difficulty of the task was also varied by using in one situation, very familiar cities, e.g. Amsterdam, London, Berlin, etc., or by using, in another situation, less well known towns to our Finnish subjects e.g. Erwitte, Hulia, Piney, etc. The major variable was the macro option. Half of the subjects had the opportunity of defining and using macros but the other half had no macro option available.

The subjects were tested individually. They made 32 navigation tasks in all. The only within-subject variable was the depth. Familiarity with cities and the macro option were between-subject variables. The tasks were presented in eight groups of four tasks of the same depth, i.e. 2 or 4. To control the learning process, the presentation was ordered so that after a task group of depth 2 came a group of depth 4 and vice versa. The presentation was counterbalanced by presenting a half of the subjects with depth 2 first and half of the subjects with depth 4 first. This was done to avoid more than four sequences of the same depth tasks.

2.3. Results

The average processing times of the subjects in different conditions are presented in Figure 1.

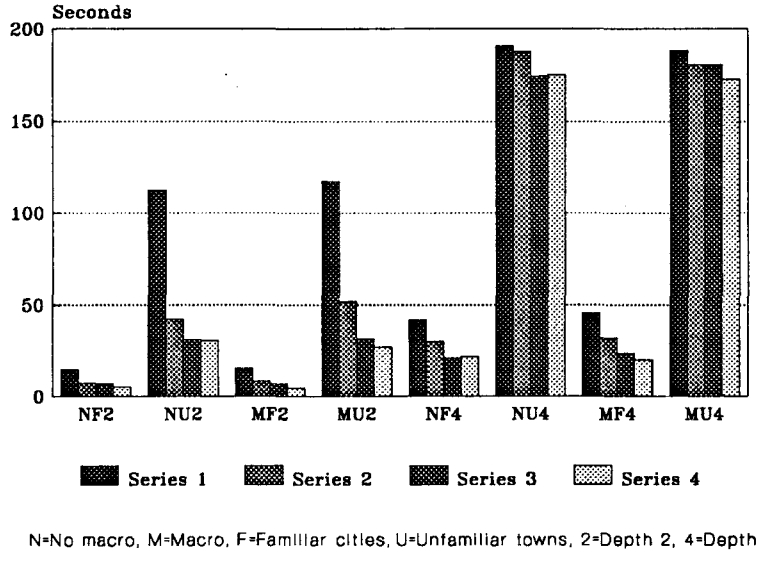


Fig. 1. The average processing times.

The main effect of task-time training, i.e., the subtask number $F(3,224)=197.44$, $p<.001$; menu tree depth, $F(1,224)=4400.58$, $p<.001$; and the familiarity of the contents, $F(1,224)=8100.18$, $p<.001$ were significant but the macro facility had no significant effect, $F(1,224)=1.40$, ns.

Also, the interactions of training and depth, $F(3,224)=40.92$, $p<.001$; training and familiarity, $F(3,224)=54.24$, $p<.001$; familiarity and depth, $F(1,224)=2234.25$, $p<.001$ as well as the three-way interaction between training, familiarity and depth, $F(3,224)=80,74$, $p<.001$ were all significant.

The frequency of the constructed macros as a consequence of tasks solved is presented in Table 1.

Table 1. Frequencies of constructed and used macros during the time of experiment by all subjects in the successive presentations of subtasks.

FAMILIAR CITIES			UNFAMILIAR TOWNS		
Learning (Series)	Macros defined	Macros used	Learning (Series)	Macros defined	Macros used
S1	1	0	S1	0	0
S2	9	3	S2	2	1
S3	15	6	S3	4	3
S4	17	8	S4	5	4

2.4. Discussion

The familiarity hypothesis was confirmed. Subjects were able to use their geographical knowledge in menu navigation. They made much more sensible search plans in familiar tasks than in unfamiliar. This means that domain knowledge is an essential variable in menu navigation. The depth effect or task-complexity effect was also substantial, as one would expect. The deeper a path, the more difficult it is for the subjects to find the correct route.

The interaction between task complexity and familiarity is understandable. The more unfamiliar the interface, the more blind and unorganized the search (Card 1984, Vandierendonck, van Hoe, de Soete 1988). This interaction has an important consequence when the action oriented interface concept is considered. The structural decisions concerning the interface modification are not independent from the domain knowledge structures. People seem to use domain knowledge analogies in constructing mental models about the structure of the interface.

To our surprise, our main hypothesis failed. Subjects did not navigate significally faster when they had the opportunity to make macros. They simply did not try to profit from this facility, which can be seen from the frequency of macros. All the subjects knew how macros could be constructed. They made a number of macros but seldom used them. Especially in unfamiliar conditions, both macro construction and macro use was modest. Since macro construction required some extra time and planning, the subjects rejected it, as shown by the low frequency of use, and therefore did not benefit from the macro facility in their task solutions. This indicates problems for our idea of user-modified and action-oriented interface, because it presupposes wide macro use.

3. Second experiment

3.1. Introduction

In the first experiment, the macro facility was of practically no benefit. People were simply not interested in using it. The reason may either be that the subjects were not willing to use macro procedures because they thought that the costs of macro construction surpassed the benefits of its use or it may be that they simply did not take into account the use of this possibility.

To test these hypotheses for the lack of use of the macro option, we decided to use computer cuing. It is well-documented in the cognitive literature that cues after beneficial effects in problem solving (Maier 1930, 1931). Also, Wetzenstein-Ollenschläger and Wandke (1990) have shown that subjects, in using an adaptive interface, may be encouraged to select more

complex-action levels by computer cuing. If the reason for the low use of macros is overlooking this facility, computer cuing should be an effective cure for the problem. If the reason is conscious avoidance of macro options, cuing cannot provide much help.

3.2 Method

3.2.1 Subjects

Thirty two male students with similar backgrounds and ages as in the first experiment were used. None of them had participated in the first experiment. The subjects were divided in four equallysized groups.

3.2.2 Materials

ACONA I was modified for this experiment by adding it a facility which actively suggested macros. The new program was called ACONA II. ACONA II monitors the paths subjects frequently use. If a path is used four times, the program interrupts, reminds subjects that they have used this route several times and asks if the subjects would like to make a macro. After a "Y/N" answer, ACONA II operates precisely like ACONA I. In ACONA II, no indication of the macro elements are needed and it defines the first and last element by the frequency of visits. If in the current path, some of the cities have a visit frequency higher than three, the program takes it as an element and backtracks until a city of the same visit frequency is found. Macro defines the route between those two cities. However, since the program never makes the same suggestions twice, the backtracking is continued until a menu element of the same frequency is found. If no suitable menu element can be found between root and the currently active element, no suggestion is made.

3.2.3 Procedure and design

The procedure was the same as in the first experiment. Design familiarity versus unfamiliarity was the between-subjects variable and depth was again the within-subjects variable.

3.3. Results

The main results are presented in Figure 2.

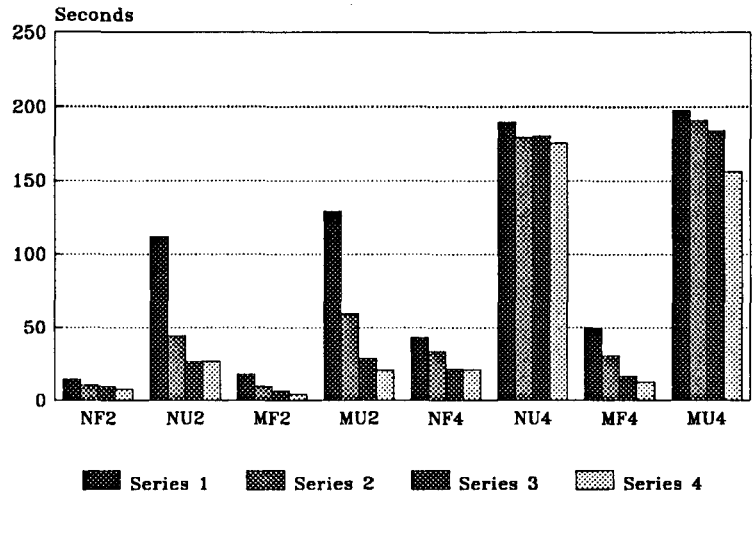


Fig. 2. The average processing times.

The following main effects and interactions were significant: task-time training, $F(3, 224)=175.87$, $p<.001$; depth $F(1, 224)=2791.29$, $p<.001$; familiarity, $F(1,224)=5302.26$, $p<.001$; training x macro $F(3, 224)=8.64$, $p<.001$; training x depth $F(3, 224)=26.17$, $p<.001$; training x familiarity, $F(3, 224)=44.74$, $p<.001$; familiarity x depth, $F(1, 224)=1528.54$,

$p<.001$, as well as the three-way interaction between training, depth and familiarity, $F(3, 224)=67.71$, $p<001$. However, the macro effect is not significant $F(1,224)=0.77$.

Frequencies of constructed and used macros are presented in Table 2.

Table 1. The frequencies of constructed and used macros during the time of experiment by all subjects in the successive four series (S).

Familiar cities			Unfamiliar towns		
Learning (Series)	Macros defined	Macros used	Learning (Series)	Macros defined	Macros used
S1	5	0	S1	3	0
S2	7	3	S2	13	2
S3	15	14	S3	18	7
S4	17	21	S4	23	17

This time more macros were constructed and used than in the first experiment. The increase in macro construction occured in tasks with unfamiliar cities, in which the total number of uses was over four times greater in this experiment than in the first. The number of macros used increased in both familiar and unfamiliar groups.

3.4. Discussion

The second experiment supports the previous results. Domain knowledge is used in navigation planning in a menu system. The increase in tree size causes more problems for the subjects. In an unfamiliar environment especially, the menu-tree depth is an essential factor. Computer cuing had no effect on these factors.

Interestingly, the macro facility interacted with task-specific learning. The means that when the number of solved tasks increased, subjects benefited from use of themacro facility, which did not happen in the previous experiment. Analysis of macro use also shows that the frequency of macro use increased with tasks pecific learning. The more familiar people are with the task, the more macros they construct and use.

In the beginning of experiments the macro index is empty and thus no macro use is possible. To learn to use macros takes some time and so does the building of the basic macro library. Therefore, the actual macro effect can take place only when the number of trials increases. This might explain why no significant macro-effect was found. In using macros, subjects may lose time in the beginning but, when the number of tasks and the basic macro storage increases, the real benefits of an adaptive interface become apparent.

4. Third experiment

4.1. Introduction

The second experiment suggested that action oriented adaptive interface has benefits only when the number of trials increases. This means that, in this experiment, the number of trials should be increased to validate the benefits of computer cuing and adaptive interface.

We also thought that increasing menu-tree depth, or task complexity, would increase the effects of macro use. Increasing task difficulty should motivate subjects to use all the available facilities to decrease mental effort. Hence, we increased the depth of the menu tree. We predicted that the benefits of interactively supervised menu modification would increase when the number of trials and the depth of the menu tree increased.

4.2 Method

4.2.1 Subjects

One female and seven male subjects were used. Their backgrounds were the same as in the previous experiment. They were divided into two group of four. None of them had participated in the previous experiments.

4.2.2 Materials

ACONA II was used. The menu tree depth was increased so that depths of three and six were used. Each menu node contained 3 alternatives. The total maximum search depth was seven and total number of menu elements or nodes in the menu tree was 191. Otherwise, the interface was identical with the second experiment.

4.2.3 Procedure and design

The procedure and design were the same as in the previous experiment with the exception of the search tree size. The total number of tasks was increased from 32 to 64.

4.3 Results

The results of the third experiment are presented in Figure 3.

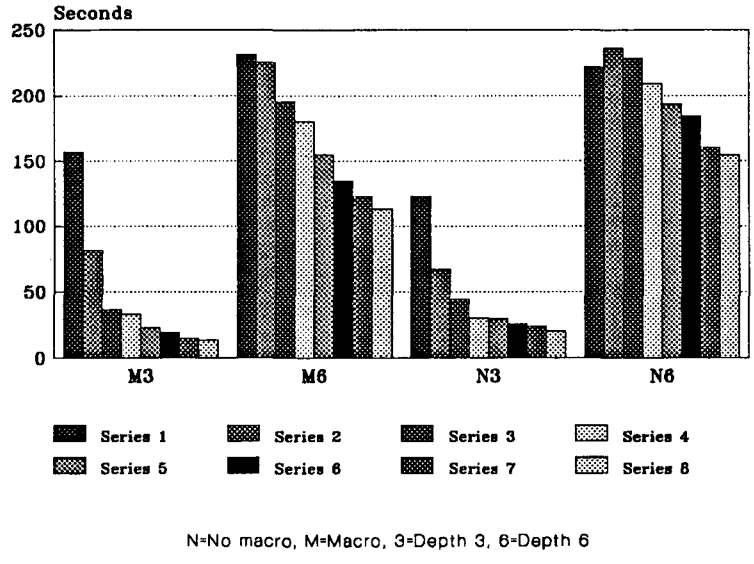


Fig. 3. The average processing times.

All the main effects of task-time training, $F(7,96)=104.39$, $p<.001$; and depth, $F(1, 96)=2840.00$, $p.<001$; were significant. The effect of macro option was significant, $F(1,96)=27.17$, $p<001$. The interactions between task-time training and macro option was significant, $F(7,96)=5.38$, $p<.001$; macro option and depth, $F(1,96)=34.82$, $p<.001$; as well as interaction between depth and training, $F(7,96)=14.69$, $p<.001$ were significant.

Table 3. Frequencies of constructed and used macros during the time of experiment by all subjects in the eight series.

Learning (Series)	Macros defined	Macros used
S1	1	0
S2	6	1
S3	7	4
S4	9	7
S5	13	12
S6	15	19
S7	17	27
S8	18	33

Although macro construction was not more frequent compared to the previous experiment in all cases, the frequency of macro use substantially increased with the increase in the number of subtasks compared to the previous experiments.

4.4. Discussion

The interactive action-oriented macro facility substantially increased subjects' use of the macros when the number of subtasks and the task difficulty increased. The total difference in the frequency of macro use is a very clear indication of this. The shortening of the time used per task was also substantial.

In the first experiment, subjects were not sufficiently willing to use the macro construction facility. They knew about the macro construction facility but they did not want to use it. The second experiment gave clear hints that computer cuing might increase macro definition and use. Finally, this last experiment showed very clearly that subjects benefit from computercued macro-construction when the number of trials increases.

We think that the basic difficulty with the non-cued macro facility was caused by apperceptive processes (see Saariluoma 1990, 1992, for the notion of apperception). When subjects have to plan their routes in the menu, their action-plan and representation does not naturally include the idea of macro construction. Planning a new macro requires restructuring and a temporary shift of attention from the main task, i.e. actual navigation. This requires cognitive costs which, at first, seemed to surpass the expected benefits.

The subjects needed to use their resources to maintain the navigation process as they did not necessarily know precisely where they were going to nor where they came from in the current node. Consequently the planning needed in macro construction without computer cuing was more difficult when they did not have sufficient knowledge about the environment. Cuing requires the subjects' to shift their attention from task knowledge in order to make a macro. In this way, cuing helps the subjects to pay attention to the environmental action instead of just focusingall their attention on the actual task.

5. General discussion

The psychology of action oriented user-interface relies on chunking. Human actions have hierarchical structures in which low-level basic actions are associated into wholes by chunking. Chunking is the way human information processing avoids the limits of working memory (Chase and Simon 1973, Ericsson and Kintsch 1991, Miller 1956). The mental load caused by acting in an environment depends essentially on human ability to chunk task-necessary environmental information. The more difficult the chunking, the more mental load the task causes and the more difficult performance will be (Gopher and Donchin 1986). Thus, the idea is to remove all the obstacles of natural chunking from an interface and to use all the technical potential in computers to foster the chunking of actions.

A problem with many computer-interface systems is that they divide actions into rigid sets of elements and, in this way, impaire the chunking of action components. If each menu during the navigation process must be separately processed during the navigation process, the natural chunking process is disturbed. People learn the standard paths in a menu system, their information collection changes, and before long, they will find it burdensome to go through the piecemeal processes again (Card 1984, Vandierendonck, van Hoe, de Soete 1988).

The current experiments showed that the use of adaptive interfaces raises a new psychological problem. This is attention. When the users try to achieve some action-goals and when their apperception has abstracted one action plan, they are not very willing to pay attention to the modification of the interface (cf. Saariluoma 1992). It is one thing to navigate in a menu system and another to modify it. Computer cuing seems to provide an effective means to solve this problem (cf. also Wenzenstein-Ollenschläger and Wandke 1991).

The more complex and memory loading an environments is, the more the users benefit from the action-oriented design. This can be seen especially in the subjects' performance in the long, unfamiliar menu-navigation tasks.

On the whole, the idea of action-oriented user-interface has been insufficiently researched. These first experiments suggest that action-oriented and adaptive user interfaces, though not being one and the same thing, have much in common. The combination outlined in this paper could be characterized action-oriented adaptive user-interface. This notion seems to have some flexibility. For example, it allows very effective personalization of the interface design. The macro systems created by one user can be stored in a personal file and thus the same interface may be modified in very different directions. The interface may be build differently by each of its users depending on their personal needs. This we find is one of the most interesting future aspects of action-oriented adaptive interfaces.

References

Barnard, P. J. and Grudin, J (1988). Command names. In: M. Helander (ed.), Handbook of human-computer interaction. Amsterdam: North-Holland. (pp. 237-255).

Card, S. (1984). Visual search of computer menus. In: H. Bouma and D. Bowhuis (eds.) Attention and performance, vol 10. Hillsdale, N. J.: Erlbaum. (pp. 97-118).

Chase, W. G. & Simon, H. A. (1973) The mind's eye in chess. In: W. Chase (ed.), Visual information processing. Academic Press: New York. (pp. 215-281).

Ericsson, K. A. & Kintsch, W. (1991). Memory in comprehension and problem solving: A long-term working memory. Institute of Cognitive Science, University of Colorado, Boulder. Publication Number 91-13.

Gopher, D. and Donchin, E. (1986). Workload - an examination of the concept. In K. R. Boff, L. Kaufman & J. P. Thomas (eds.), Handbook of perception and human performance. Vol. II: Cognitive processes and performance. New York: Wiley.(pp. 41/1-49).

Miller, G. E. (1956) The magical number seven plus or minus two: Some limits on our capacity for processing information. Psychological Review, 63, 81-97.

Murray, D. (1991). Modelling for adaptivity. M. J. Tauber and D. Ackermann (eds.), Mental models and human-computer interaction, vol II. Amsterdam: Elsevier.(pp. 81-95).

Maier, N. R. (1930). Reasoning in humans I: On direction. Journal of comparative psychology, 12, 181-194.

Maier, N. R. (1931). Reasoning in humans II: The solution of a problem and its appearance in consciousness. Journal of comparative psychology, 12, 181-194.

Mäntylä, T. (1986) Optimizing cue effectiveness: Recall of 500 and 600 identically learned words. Journal of experimental psychology, 12, 66-71.

Paap, K. R. and Roske-Hofstrand, R. J. (1988). Design of menus. In: M. Helander (ed.),Handbook of human-computer interaction. Amsterdam: North-Holland.

Vandierendonck, A., van Hoe, R, and de Soete, G. (1988). Menu search as a function of menu organization, categorization and experience. Acta psychologica, 69, 249-278.

Wetzenstein-Ollenschläger, E. and Wandke, H. (1990). Toward adaptive human-computer interfaces, In:D. Ackermann and M. J. Tauber (eds.), Mental models and human-computer interaction, vol I. Amsterdam: Elsevier. (pp. 231-252).

Peter Mikulecký
Department of Artificial Intelligence
Faculty of Mathematics and Physics
Comenius University Bratislava

Title:

Active knowledge-based interfaces for existing systems

Abstract:

The complexity of computer exploitation has increased rapidly as computers became components in complex systems. This is true even if we consider software systems or packages. To use them meaningfully requires days (even weeks) of practicing with necessity to study thick manuals. This seems to be especially critical in the case of expert systems (but not only), where the user expects to begin useful work almost immediately. The moral of this is that computer users, regardless whether they program or not, expect *more and more assistance from the systems* they use.
We do believe that it is the *idea of active (or co-operative) assistance* to computer users, which seems to be very promisful and essential for designing *intelligent interfaces* (or front--ends) to existing complex software systems. In the contribution, we intend to present a theoretical background about *active assistance*, including the basic problems from the area of *advice giving*. Next, some *recent projects of active advice-giving systems* or other intelligent interfaces will be reviewed and some common problems will be discussed. Some attention will be paid also to the problems of *restricted natural language facilities* in knowledge-based front-ends.

Intelligent Knowledge Based Systems in Intelligence Analysis

Konrad Morgan & Laura Holland,

**Department of Information Science,
University of Portsmouth.**

Phillip Hardy, James Casey, Tony Quinn & Roger Mead,

**Information Technology Branch,
Sussex Police.**

Keith Roberts & Ian Cadwell,

**Police Systems,
McDonnell Douglas Information Systems.**

Richard Oldfield,

**The Home Office,
Police National Computer Organisation.**

KEYWORDS : Human-computer-interaction; Intelligent-knowledge-based-systems; Object-oriented-interface-design; Criminal-intelligence-analysis; Computer-supported-cooperative-work;

ABSTRACT. The paper describes some of the encouraging possibilities which are emerging in the area of computer based criminal intelligence analysis. The project described in the paper has investigated current methods of criminal intelligence analysis and the problems which confront police services in the investigation of serious crime. Using simulated crime scenarios these problems are illustrated along with proposed solutions and improvements which are possible with modern computing techniques.

The project(5,6), named Mycroft^{TM} after Sherlock Holmes' genius brother, is a collaborative effort between the Sussex Police , McDonnell Douglas Information Systems, The UK's Home Office, and the Department of Information Science at the University of Portsmouth.

1.INTRODUCTION.

The field of computer based criminal intelligence analysis is a domain of computing which has received relatively little academic attention (1,2). A review of the past 25 years of general computing and the HCI literature shows little mention of the many unique and challenging computational problems encountered in criminal intelligence analysis. Yet there can be few more deserving areas than aiding the solution of crime with its adverse impact on the standard of life and freedom enjoyed by the general population. In recent years the amount of data involved in the investigation of major crime, such as murder, has increased dramatically (1,3,4). Police forces world wide are starting to find that traditional methods of criminal intelligence analysis which had served them well in the past are now struggling to cope with the increasing amounts of data involved in modern criminal investigations (4).

1.1.PRESENT WORKING METHODS IN CRIMINAL INTELLIGENCE ANALYSIS.

The typical criminal intelligence investigation involves many of the problem domains which are currently being explored by both the human-computer-interaction and artificial intelligence communities. The nature of the work in major incident rooms involves real-time data analysis, cooperative working, and shared data-base & knowledge-base manipulation. Typically current techniques include the use of large computer based data indexing systems and single media data representations, such as wall charts and informal information exchanges (1,4).

1.2.INTERNATIONAL VARIATIONS.

It is important to recognize that while different countries adopt variations in dealing with the problem of investigating major crime these differences are often due to variations in the legislation which constrains the methods of operation (such as civil rights) available to the criminal investigator. Whatever the legislation, or lack of it, the task of criminal intelligence shares some fundamental characteristics.

2.FUNDAMENTAL CHARACTERISTICS OF CRIMINAL INTELLIGENCE ANALYSIS.

At the most primitive level all criminal investigations involve six stages and it is normal for an investigating team to be involved in all of these activities simultaneously.

2.1.THE SIX STAGES OF CRIMINAL INTELLIGENCE ANALYSIS.

DATA GATHERING

This may be from a wide variety of sources and be of varying degrees of reliability.

DATA ACCUMULATION

The data is centrally accumulated to ensure that all the gathered data is available to all the investigating team.

THE TRANSFORMATION OF DATA TO INFORMATION

Data of unknown reliability or relevance is transformed to useful and reliable information.

THE TRANSFORMATION OF INFORMATION TO KNOWLEDGE

Information on all aspects of the crime is transformed to knowledge based models of the events and activities which occurred. We define useful and usable information as knowledge. It is important to note

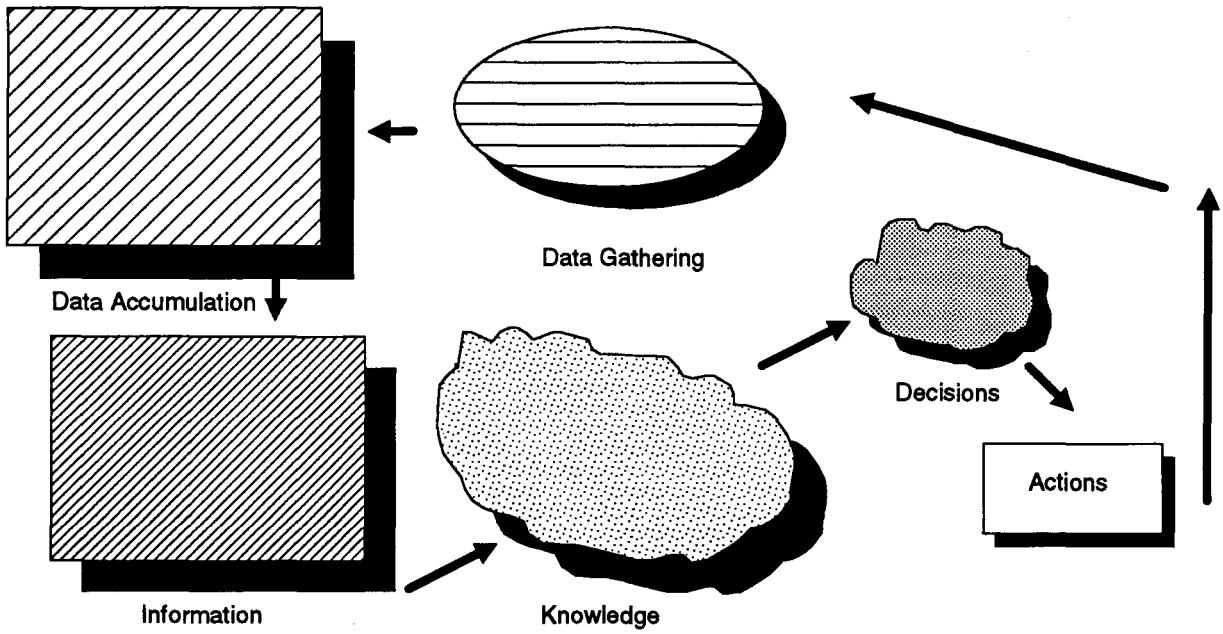


Figure 1. The Six Phases of Intelligence Analysis

that something can be information about a crime but may still not be incorporated into a knowledge base because it is currently useless.

THE TRANSFORMATION OF KNOWLEDGE TO DECISIONS

The knowledge based models are used to suggest possible activities or scenarios which may increase the strength (degree of certainty) within the knowledge base and reduce any gaps or weaknesses in the current state of the knowledge base.

THE TRANSFORMATION FROM DECISIONS TO REAL-WORLD ACTIONS

Based on the decisions suggested by the knowledge base various real-world actions are instigated.

3.CRIMINAL INTELLIGENCE KNOWLEDGE ANALYSIS TECHNIQUES AND DATA REPRESENTATIONS.

All information within a criminal investigation have some fundamental properties which are invariant. These properties can be summarized into what we will term spatial, temporal, and entity based characteristics. The combination of relationships between these characteristics theoretically enable a system, be it human or computer based, to store and manipulate any event or interaction. The more accurately and efficiently these events can be stored and manipulated the more likely it is that the information base can be efficiently modeled and transformed into a knowledge base.

3.1.EXAMPLES OF CRIMINAL INTELLIGENCE ANALYSIS TECHNIQUES TO BE EVALUATED IN THE DEVELOPMENT OF THE MYCROFT SYSTEM.

DATA SEQUENCING.

The sequence in which detectives link items of criminal intelligence data.

DIFFERENTIAL RATING OF DATA IMPORTANCE.

The criteria used to evaluate the importance of data items and inter-data-item links.

FORMALISE INTERFERENCES.

The high level inferences made by detectives from such data links.

IMPROVING THE VALUE OF WITNESS DESCRIPTIONS.

Using the known relationships between witness descriptions and the actual descriptions of detected criminals the system will try to take into account factors which might influence the witnesses description.

Example factors which might influence the witnesses description.

- Time of Day
- Environmental Light levels
- Environmental Sound Levels
- Characteristics of the Witness.
    - *Height*
    - *Size*
    - *Witnesses Work & Environmental Background*

OFFENDER PROFILES.

Assign probability ratings to possible offender profiles within the current investigation. It is hoped that this may enable the intelligent interface to automatically trawl for suspects among the information base.

CRIME PROFILES.

Assign success probabilities to suggested lines of inquiry. This decision support will rate the likelihood of success given the current status of the knowledge base.

4.MINI CRIME SCENARIO.

In order to highlight the stages and transformations which we have just outlined, and the problems which can be associated with them, we will attempt to present a simplified crime scenario. The examples produced will also allow us to present some of the possible additional assistance which may be provided by computer based criminal intelligence analysis tools.

4.1.THE CRIME.

» At 14:30 on Monday the 1st of March 1993 AverageTown police station receives a call from distressed young female, a Miss Rachel Black, who reports having found her 50 year old mother, Mrs Black, dead in their family home at number 50 East Street, Average Town.

» At 14:36 a police squad car arrives at the scene and confirms the report that Mrs Black has been found dead in suspicious circumstances. The attending officers also report that on their way to the scene, up South Street, they saw a well known confidence trickster, Fred Bogus, in the south street BT phone box.

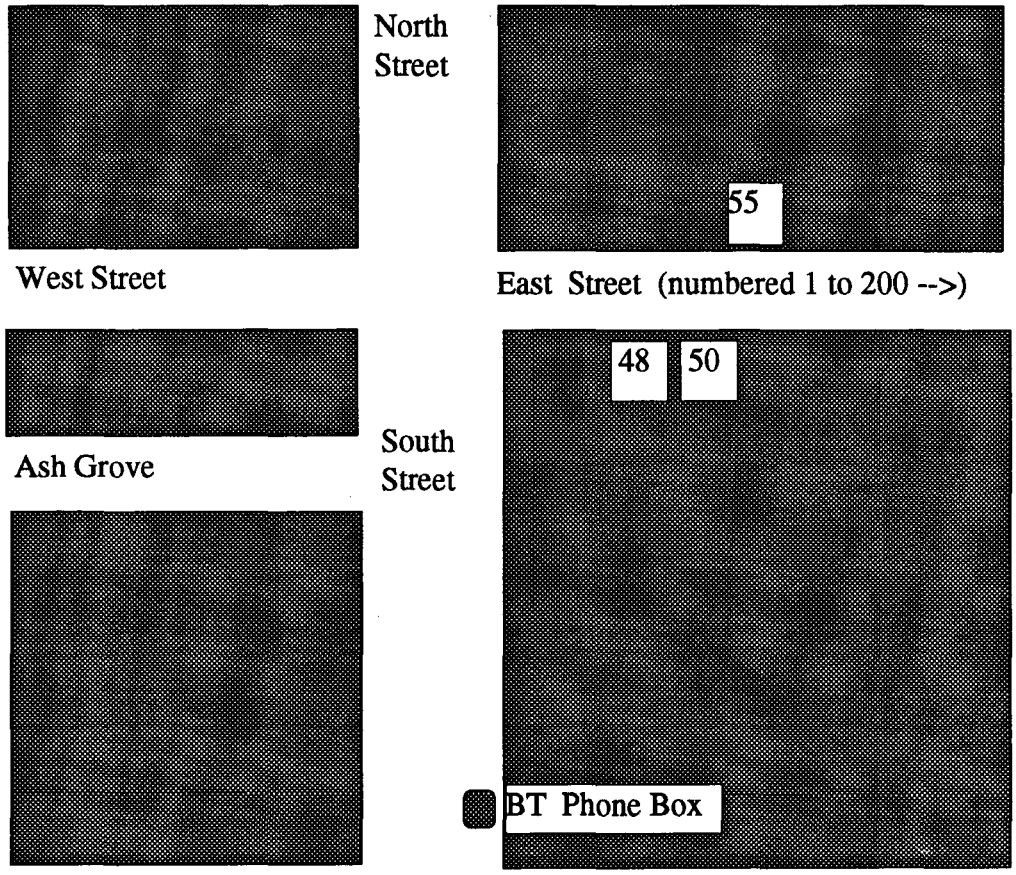


Figure 2. Main Street Plan of AverageTown

- At 14:45 additional officers secure the scene of the crime and detailed investigations within the scene take place. Miss Rachel Black is interviewed and she reports last seeing her mother alive at 13:00 hrs earlier that same afternoon. Analysis of Mrs Black's personal diary show that she was expecting a visit from her financial advisor, Mr Sterling, at 14:00HRS that same afternoon.
- Examinations at the scene of the crime show that a valuable stamp collection and a large sum of cash, which Mrs Black was hoping to invest with her financial consultant, is missing.
- House to House enquires in the local area reveal that an elderly next door neighbor of the victim, Mrs Withers, saw a very tall young man with a beard visit Mrs Black's house between 13:30HRS and 14:30HRS.
- Other residents in the street report that a door to door salesperson, Mr Dodgey, called on the houses in East street between 10:00HRS and 14:00HRS.
- A member of the public who was walking their dog reported seeing a maroon hatchback drive at high speed down the heavily shaded Ash Grove from South Street at 14:15HRS on the afternoon in question.
- One of Mr Dodgey's customers of that day, at number 55 East Street reports buying a yard brush from him at 14:00HRS.

» The incident room is set up at AverageTown Police Station and the detectives set up interviews with Miss Rachel Black, Mr Bogus, Mr Dodgey and Mr Sterling.

WE CAN NOW SUMMARISE THE DATA ABOUT EACH INDIVIDUAL INTERVIEWED.

Miss Rachel Black : 21 Years old. 5 foot 5 inches tall. Student at AverageTown Art College. States that she left her Mum to go shopping on the afternoon in question.

Mr Bogus : 30 Years old. 5 foot 6 inches tall with long beard and hair. Refuses to answer any questions.

Mr Dodgey : 35 Years old. 5 foot 10 inches tall. Sideburns and medium hair. Reported that he was doing his rounds that day but that he never got as far as Mrs Blacks at number 50. States that he parked his Red Sierra XR4i outside number 70 East Street during his sales round and after completing his sales at 14::18HRS he drove away down East Street heading in an easterly direction.

Mr Sterling : 45 Years old. 5 foot 8 inches tall. Large Moustache and slight sideburns. Receding hair. States he visited Mrs Black as arranged but was 20 minutes late (14:20HRS) got no reply and proceed to his next appointment in nearby North Street.

4.2.DEMONSTRATING THE PROBLEMS INHERENT WITHIN THE SIX STAGES OF CRIMINAL INTELLIGENCE ANALYSIS.

In the preceding simplified description of a crime we have attempted to incorporate features of the data which will allow us to show the problems facing the police service when they try to recreate and analyse the events which were associated with a crime.

Before we show some of the advanced computer supported tools and techniques which we hope to evaluate in our project we will briefly summarise some of the problems faced by investigators using current methods and techniques.

SIZE OF ACTUAL AND POTENTIAL DATA SETS MAY BE OVERWHELMING.

In our mini-crime we deliberately decided to present a small number of events, places and individuals. However we hope that even this small number of data items will have allowed the reader to start to appreciate the difficulty of accurately constructing a simulation of the events associated with a crime.

Example from the Mini Crime.
Our crime contains only a few dozen data items, while real data sets may approach 20,000 items of information.

RELEVANCE OF THE DATA IS UNKNOWN.

Not only is the data set large but the investigators have no way of knowing in advance which items of data are relevant. If you appeal to the public for information about events associated with an approximate time or place you will inevitably get large amounts of data which has little or no relevance to the crime being investigated.

Example from the Mini Crime.
There is no way of knowing if the sighting of Mr Bogus, a well known confidence trickster, is at all relevant to the crime under investigation.

DATA SET IS INCOMPLETE.

No matter how well an investigation is conducted the data model of the crime will always be incomplete. This problem is made worse because it is only with hindsight that the investigators know for certain where the most important gaps in their data exist.

Example from the Mini Crime.

Large gaps exist in our knowledge of events at the scene of the crime, 50 East Street, AverageTown between 13:30 and 14:30 HRS on the day of the crime.

DATA SET IS INACCURATE.

If two individuals recall a shared event it is unlikely that their accounts will match each others exactly - unless they have been rehearsed or prompted in some way. It is even more unlikely that their recollections will perfectly match those which actually took place. Quite unintentionally individual biases and interpretations will colour descriptions of events. These inaccuracies can produce conflicts and errors within any criminal intelligence event simulation.

Examples from the Mini Crime.

It is uncertain how accurately the elderly neighbour, Mrs Withers, described the visitor who she saw visiting the victim. If Mrs Withers was extremely short, or had developed a slight stoop, she may have overestimated the visitors height. An additional problem is what Mrs Withers described as a beard may be described by another person as 'sideburns and a moustache'.

Another example from our mini crime is the colour and description of the car which was seen driving at high speed down the heavily shaded Ash Grove. Environmental light levels will influence the perception of colour.

DATA SET IS AMBIGUOUS.

As a result of the inevitable inaccuracies held within a typical data set there may be areas of the data based crime simulation which has two or more possible mini-scenarios or interpretations. These ambiguities can only be clarified by gaining more definitive data about those data items. In the worst case these ambiguities can lead to many quite different crime scenarios which, until more or better intelligence is available, must be regarded as being equally likely.

Example from the Mini Crime.

The combination of witness descriptions makes it difficult to know if the visitor described by the elderly Mrs Withers was the financial consultant Mr Sterling, the door to door salesperson Mr Dodgey, the confidence trickster Mr Bogus, or another individual as yet unknown to the investigation.

DATA SET MAY BE MISLEADING.

A series of intentional or unintentional errors in the data provided to the form the data based crime simulation may make the resulting simulation misleading. This can lead to wasted investigations or in the worst case, a miscarriage of justice.

Example from mini crime.

Mr Dodgey's statement specified that he had not completed his door to door round in East street as far as the victim, Mrs Black's, at number 50. However one of Mr Dodgey's customers of that day lived at number 55 East street beyond Mrs Blacks, and Mr Dodgey's car was parked outside number 70 East Street. This suggests that either Mr Dodgey or the customer at number 55 has made an inaccurate statement.

INVESTIGATIONS SUBJECT LIMITED RESOURCES.

Another problem which faces the police service in their criminal investigations is that of limited resources. In some cases this can mean that the officers in charge of an investigation have to select one of many possible avenues of investigation.

Example from mini crime.
We have not actually stated what resource constraints apply to our mini crime but it is likely that the investigating officers would have limited resources available to assist them in their investigations.

4.3.EXAMPLES OF POSSIBLE TOOLS AND TECHNIQUES TO BE EVALUATED IN THE MYCROFT PROJECT.

Having, very briefly, considered some of the problems which face the police service in their unenviable tasks of criminal intelligence analysis we can now devote the remainder of this paper to providing selected examples of the various computer based tools and techniques which we intend to evaluate in the process of the Mycroft project.

DATA GATHERING & DATA ACCUMULATION

The Mycroft system will require that data items are categorised into various types. In our mini crime these categories would include; a list of identified people, a list of unidentified people, a list of places, a list of items, a list of animals involved, and a list of vehicles involved.

LIST OF IDENTIFIED PEOPLE.

- » Mrs Black
- » Miss Rachel Black
- » Fred Bogus
- » Mrs Withers
- » Mr Dodgey
- » Mr Woof-Walker
- » Mr Buy-Anything

LIST OF UNIDENTIFIED PEOPLE.

- » Visitor to Mrs Black. Male with facial hair who is taller than Mrs Withers.

LIST OF PLACES.

- » Ash Grove, Average Town
- » BT Phone Box, South Street, Average Town
- » 48 East Street, Average Town
- » 50 East Street, Average Town
- » 55 East Street, Average Town
- » 70 East Street, Average Town

LIST OF ITEMS.

- » Mrs Black's Personal Diary
- » Valuable Stamp Collection
- » A Large Sum of Cash

LIST OF ANIMALS INVOLVED.

- » Large Dog.

LIST OF VEHICLES INVOLVED.

- » Maroon Hatchback
- » Red Ford Sierra

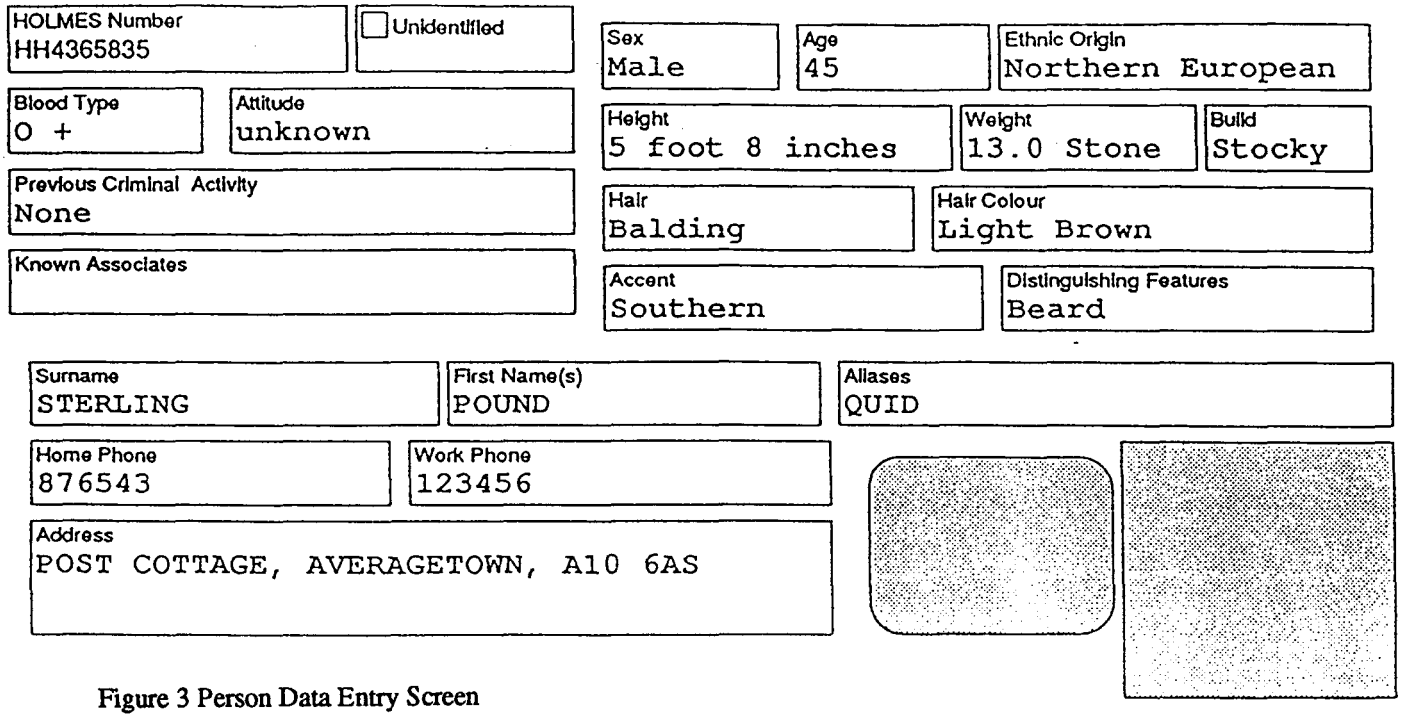


Figure 3 Person Data Entry Screen

DATA ENTRY.

Each data category has its own set of computer based data entry templates (see figures three & four) which in addition to aiding accurate and efficient data entry also build up a table of logical flags which are used by the embedded expert system in its various knowledge based operations. Through a detailed six month study (2) we created and evaluated a prototype of these logical tables using flags such as temporal & spatial proximity to the crime, likely accuracy of the entry (witness validity), witness characteristics, uniqueness of the item, number of

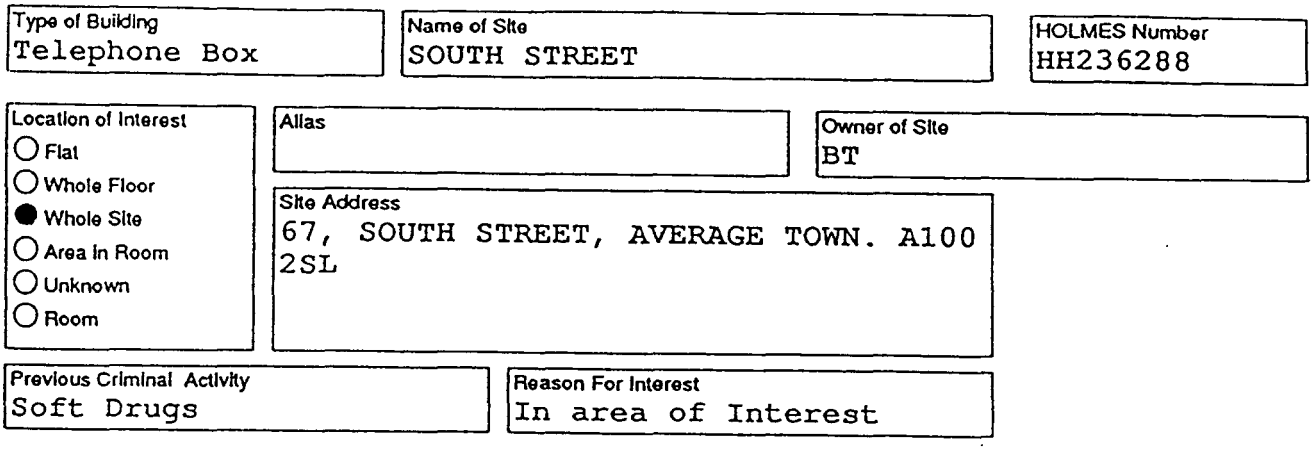


Figure 4 Data Entry for a Place

times accessed or cross-referenced and the age of the data.

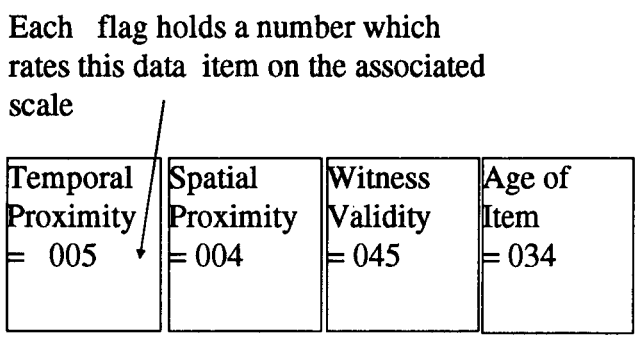


Figure 5 Example Logical Data Flags

THE TRANSFORMATION OF DATA TO INFORMATION

By means of these logical flags, and the stored data, the intelligence analysts can begin to assess the importance and relevance of the gathered data. It is hoped that the Mycroft system will concentrate the attention of the intelligence analyst to those items of data which provide valuable information about the crime.

THE TRANSFORMATION OF INFORMATION TO KNOWLEDGE.

The Mycroft system will support various novel ways of simulating the known events that are associated with the crime. The easiest method of demonstrating the potential of these techniques is to show how the Mycroft system might assist in various example mini-crime problems we discussed in the preceding section (above).

HIGHLIGHTING ONLY THE RELEVANT.

By specifying a spatial background within the area of the crime a digitised map of the area is presented. The analyst can then specify the time period which he wishes to investigate and only data elements which occurred within the specified spatial and temporal limits will be displayed. It is equally possible to view the same set of information with temporal and spatial axis exchanged, thus allowing unique perspectives on the data.

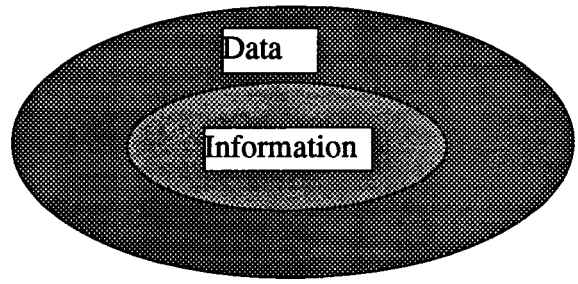


Figure 6. An Information 'Window' on the Raw Data

HIGHLIGHTING GAPS IN THE KNOWLEDGE BASE.

Using the different information views, which we have just described, the Mycroft system will also highlight those areas in the knowledge base which contain too little information.

HIGHLIGHTING INACCURACIES.

Since each data item appears as an icon within the current view inaccuracies are more easily visible. In addition the expert system will search for and highlight any conflicts which may be within the data set. For example in our mini crime Mr Dodgey's statement that he did not reach as far as number 50 East Street would be highlighted by the system showing his icon visiting number 55.

HIGHLIGHTING AND HELPING TO COPE WITH AMBIGUITY.

The Mycroft system would highlight the possible problems surrounding Mrs Withers' description of the visitor to Mrs Black. Mrs Withers' description could match any of the male suspects. However Mycroft's knowledge base would recognise that Mrs Withers is short and would therefore tend to over estimate height.

Another example is where Mr Woof-Walker described a maroon coloured hatchback travelling at speed down the shaded Ash Grove. Mycroft would highlight the possibility that because the car was so close (temporally and spatially) to Mr Dodgey's departure, the car in question could be Mr Dodgey's Sierra traveling in a Westerly direction.

THE TRANSFORMATION OF KNOWLEDGE TO DECISIONS & REAL-WORLD ACTIONS.

HELPING TO MATCH RESOURCES TO THE INVESTIGATION.

The Mycroft system will record the actions which are generated and the resources used in the course of the investigation. It is hoped that this feature will allow the investigating officers greater control and feedback in how their investigation is progressing.

BACKGROUND PROCESSING.

In addition to the various analysis support features we have already mentioned the knowledge base embedded within the Mycroft system will be capable of independent background processing using expert systems derived from the experience of expert detectives, the use of historical crime and offender profiles. Over periods when the system is not being used to its full processing power it will be able to conduct its own analysis of the data sets with the goal of generating a set of recommendations which will be ranked on what is judged to be their potential outcome.

5.CONCLUSIONS.

The Collaborators Expectancies.

Law Enforcement.

The successful conclusion of the project will leave the police with a tool that will have major benefit in the investigation of serious crimes. Further the force will have developed considerable experience in creating "expert" systems. It is intended that these areas be capitalized to further expand the role of Mycroft into other areas of police work. For the techniques it employs could equally be used in formulating organizational policy and other areas where the relationships between entities need to be simulated or modeled, and where intelligent knowledge based systems could be employed.

Academic.

The academic side of the Mycroft project will form a contribution to the advancement of knowledge in the following specialist areas of computer science : object oriented design & programming, knowledge elicitation in complex and vague domains, simulation of human decision making on 'complex' and 'fuzzy' data sets, design and creation of an IKBS for criminal intelligence work, human computer interface design & evaluation, technology attitudes, the impact of the introduction of new technology and methods : both in the police force and in society.

Commercial.

McDonnell Douglas see the Mycroft project as having the potential for developing into an ongoing collaborative research programme between the existing project partners. It is hoped that this collaboration would continue to develop through subsequent generations of the Mycroft system and its IKBS rule base.

Government.

The scientists within SSG are interested in the development and enhancement of existing and new intelligence analysis methods and tools. The continued development of Mycroft beyond the stages outlined in this document would ensure that the very latest tools, methods and techniques were available to the UK's criminal intelligence taskforce.

5.1.CONCLUSION.

We have invested a considerable amount of effort and resources into what are new and challenging areas of human-computer-interaction and artificial intelligence. So far the Mycroft project has established a new and unique method of intelligent multi-media information and knowledge representation and presentation. This promises to provide a truly adaptive method for knowledge representation and manipulation. We invite other parties to join us in this exciting area of inquiry.

6.REFERENCES.

1 - Hardy, P. (1991) "Information Vs Knowledge : An Alternative View of Major Incident Rooms" Internal Paper : Sussex Police.

2 - Holland, L. (1991). "An Investigation of Knowledge Elicitation Techniques in Criminal Intelligence Settings" Unpublished MSc Dissertation, University of Portsmouth.

3 - Oldfield, R.W. (1988) "The Application Of Criminal Intelligence Analysis Techniques To Major Crime Investigation - An Evaluation Study". Scientific Research And Development Branch. Home Office Publication 30/88.

4 - Woods, A.J., Jenkinson, R.E. (1991) "Intelligence and Operational Systems HOLMES Front End Demonstrator Evaluation Report " Scientific Support Group Report, Home Office Police National Computer Organisation Publication 1/91.

5 - Morgan, K., Holland, L., Hardy, P., Casey, P., Quinn, T., Mead, R., Roberts, K., Cadwell, I., Oldfield, R. (1993) "A Description of the Possible Role of Computer Enhanced Criminal Intelligence Data Analysis" Paper Presented to the British Psychological Society's Division of Criminalogical and Legal Psychology Annual Conference, Harrogate, 1st March 1993.

6 - Morgan, K., Hardy, P., Casey, J., Holland, L., Quinn, T., Mead, R., Roberts, K., Cadwell, I., & Oldfield, R. (1992) "Implementation and Design Issues in Cooperative Knowledge Based Systems for Criminal Intelligence Analysis : The Mycroft Perspective". Paper presented to the Cooperating Knowledge Based Systems Special Interest Group (CKBS) Meeting at Queen Mary & Westfield College, University of London. December 9th 1992.

Learning in Neural Networks

Paul Munro

Department of Information Science, University of Pittsburgh, Pittsburgh PA 15260 USA

Abstract. Until recently, the overwhelmingly dominant mode of machine computation has been absolutely deterministic. Neural networks provide an alternative approach to computation that is inspired by neurobiological principles. These systems can learn categories from examples, and generalize their learning, such that novel examples can be classified. Correct classification depends on a variety of factors, including the goodness of the sample used for training and the consistency of the test item with the statistics of the training set. Neural network training procedures are more closely related to statistical regression techniques than they are to mainstream AI. Analyses of the networks both during and after training show remarkable similarities to human learning, and may give insight to the principles underlying human information processing.

1 Background

The biological principles that enable higher level cognition are just beginning to be understood. While the neural components that make up our brain are so much slower than the electronic devices found in computers, they are much more numerous, and thus process information in parallel; this massive parallelism gives rise to forms of computation that are very different than those performed on electronic computers, which process instructions one at a time in sequence. Parallel distributed processing has been found to account for several cognitive phenomena (see Rumelhart and McClelland; 1986) from perception (e.g., optical illusions) to higher levels (e.g., language and problem solving).

1.1 Neural Network Fundamentals

The theory behind parallel distributed processing borrows its essential concepts from neurobiology, but a host of aspects considered vital by neurobiologists are ignored (to the chagrin of many). Of course, no singular description describes all neural network models, and so the following description should not be interpreted as a strict definition. A neural network is defined as a structure consisting of nodes joined by "one-way" connections (i.e., a directed graph). Each node receives signals along connections that terminate on it, and transmits a signal on connections that it originates. The transmitted signal is a function of the received signals and some internal parameters. A small network is shown on the left of Figure 1.1, and a general node is shown on the right, where the output is a function of the inputs, s, t, u, etc., and some internal parameters, A, B, C, etc.

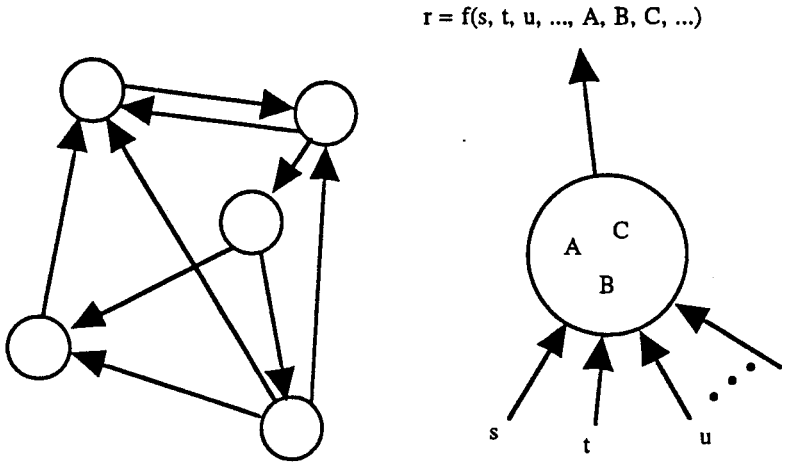


Fig. 1.1. A small neural network (left) and a schematic of one node (right).

The internal parameters of the nodes are typically weight factors assigned to the node's input signals. So in Figure 1.1, the unit might have a response like r=As+Bt+Cu. More typically, this weighted sum would be computed and then passed into a threshold-like function, like r=0 if (As+Bt+Cu)<0, and r=1 otherwise. Hence each node produces an output value (r) in response to a "stimulus pattern" of input values (s, t, u). Information is presented to the network and read from it by specifying two subsets of nodes, respectively known as the input and output layers. For a given network architecture, the output is ultimately a function of the input values (to which the input nodes are assigned), and the weight parameters of all the nodes. Thus, proper functioning of a network requires appropriate parameter values.

Neural network models can be partitioned into those that are trainable (modifiable weights), and those that are not (fixed weights). Fixed weights are generally assigned according to principles that relate to presumed correlations between nodes. In these models, each node represents a hypothesis; consistent hypotheses are connected by positively weighted connections and inconsistent nodes have negative connections, with the weight magnitudes reflecting the degree of correlation (or anti-correlation). The notion of correlation motivated an early principle for how the strengths of the synapses (the biological "connections" between neurons) are modified. Hebb's (1949) "neurobiological postulate" has inspired many mathematical approaches to learning in neural networks (e.g., Hopfield, 1982):

> *"When an axon of cell* A *is near enough to excite a cell* B *and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that* A's *efficiency, as one of the cells firing* B *is increased."*

The ability to train networks by example has captured the attention of the research community. Consider a classification task, for example, in which a population of input nodes corresponds to various measurements or observations, like light intensities on the retina, or phoneme sequences, and several output nodes that represent potential categories to which the input stimulus might belong, like "images of trees", or "the word, 'appetite'". Training generally involves the following notion: given a set of known data pairs, the weight parameters are initialized to random values, and then iteratively modified such that each stimulus in the set produces the appropriate response. After this learning procedure, the network has hopefully abstracted the relevant features of the input that are used in the classification task. If so, the network will respond appropriately to novel stimuli (i.e. data not in the training set). Thus the procedure is related to the idea of estimating a function from a finite collection of sampled data. Regression, the statistical tool for fitting functions to data, is reviewed below; as we will see, neural network training is fundamentally a kind of regression.

1.2 Regression

Since the dawn of modern science, researchers dealing with data have recognized the need for procedures that can generalize from examples. Regression techniques are the most common for this kind of analysis. In broad overview, regression requires two steps: [1] Assumption of a (parametrized) functional form that the data will be assumed to follow, and [2] optimization of the parameters for the given data. The neural network training approach described in this article is a kind of iterative regression technique, in which the class of functions is inspired by ideas of neuronal processing and architecture.

The example of linear regression is most commonly used, in which we have a set of data pairs that are assumed to be be related in a way that is reasonably approximated by a linear relationship. Given the data pairs, the regression technique can be used to find the straight line, that comes closest to passing through the data points. Simple mathematical relationships (linear, exponential, etc.) between measurable quantities are very rarely followed exactly, but under appropriate conditions, they are often close approximations.

Of course, many data sets can be more closely approximated by functions that are nonlinear. In general, the more parameters used to specify a function, the more powerful the regression; that is, the space of possible functions is more extensive. However, too many parameters leads to "overfitting" the data, a situation in which the known data is precisely fit by a high order function, while a lower order function would generalize better even though it performs imperfectly on the training data. This is particularly a problem when the data is imperfect due to errors or imprecise measurements (generally unavoidable). Figure 1.2 shows how a high-order function can be fit *precisely* to data that is roughly linear, but generalize in ways that are absurd. Linear regression is predicated on the assumption that the variables

x and y are linearly related and stipulates a formula for the finding parameters a and b to fit the general linear relation $y = ax + b$ that most closely approximates the data. However, if there are more than two data points, the regression line will not exactly intersect the data points exactly, in general. A function with more parameters can fit the data precisely. For example, a polynomial function of order n can generally be found to precisely fit n +1 data points. The quality of generalization tends to diminish as n increases (see the dotted lines and arrows in Figure 1.2).

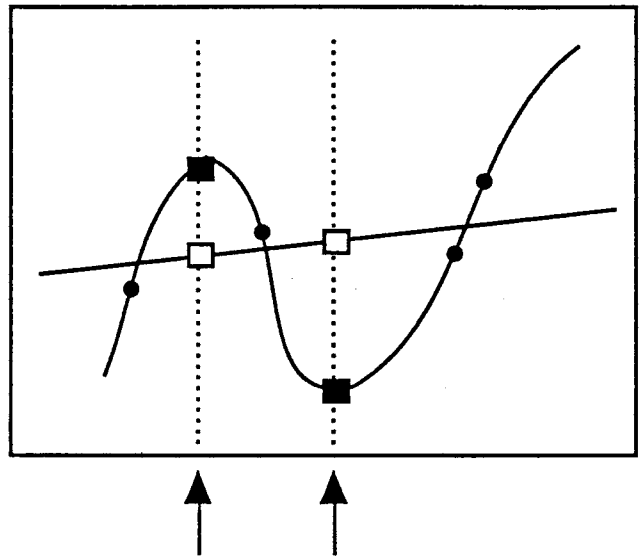


Fig 1.2. Four data points (filled circles) are fit approximately by a straight line and exectly by a third order polynomial. Subsequent approximations to two test values (arrows) by the linear fit (open squares) and the cubic curve (filled squares) are not in agreement.

In certain cases, like linear functions, the function parameters can be computed directly from the training data. In the absence of a direct formula (which is generally the case), an iterative process can be used. The strategy is to start with random values for the parameters, which will give a poor fit (barring extreme good fortune). Then, a single item (x,y) from the training set is checked against the random curve, by computing the "error", y - f(x; a,b,...), and the parameters a, b, etc. are altered to reduce this difference. After repeating this process many times for the entire training set, the function will hopefully reach an acceptable approximation to the data, and the further hope is, of course, that the function will generalize well. This general approach of *error correction* has been found to be broadly applicable in neural networks and other adaptive systems. However, error correction schemes do not always lead to the optimal set of parameters. The problem stems from the gradual nature of changing the parameters. Since only small changes are permitted, the procedure can get stuck in a region of the parameter space where a locally optimal point is "surrounded" by a neighborhood that gives higher errors for every small

change in the parameters, so the parameters cannot change to a more distant region where the error may be lower. This effect, known as a "local minimum", is illustrated in Figure 1.3.

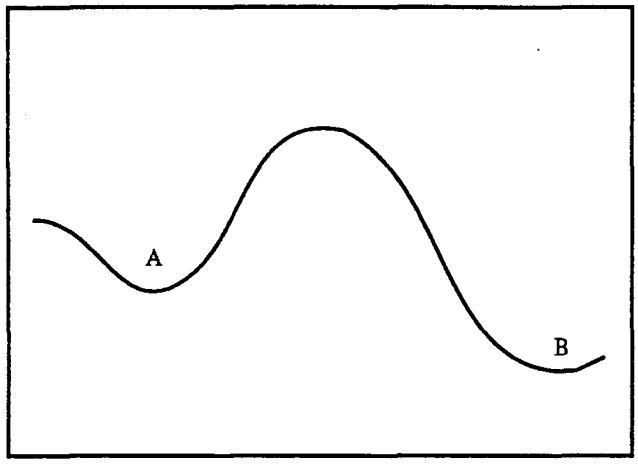


Fig. 1.3. An error function for a hypothetical one-parameter system. An error-correction procedure could get stuck at point A, a local minimum, since small changes in x do not reduce the error. Hence better parameter values, like B, will never be found. Note that the success of this procedure depends on the choice of the initial parameter value.

2 The Perceptron and Learning

Several techniques for training neural networks have been put forward, most based on either regression techniques, or Hebb's postulate, or both. Of these, one has emerged as the dominant candidate. The idea was first sketched by Rosenblatt (1961), but he was not able to implement it. The impasse was not overcome until the work of Werbos (1974), whose contribution remained unrecognized for over a decade, when three papers were independently published that showed how Rosenblatt's idea could be implemented and gave hints as to its implications for machine learning and cognitive science (Le Cun, 1986; Parker, 1985; Rumelhart, Hinton, and Williams, 1986).

2.1 Rosenblatt's Perceptron

A simple procedure, the "Perceptron Learning Procedure", to build a neuron-like classification device was devised by Rosenblatt (1961). Using this scheme, a linear threshold unit is trained to learn a categorization task by example. Here, we assume that there are two kinds of stimulus patterns, those in the category (A), and those outside (B). A perceptron unit responds with either a 0 or a 1; optimal performance

on the task corresponds to responding with a 1 to all the patterns in A, and with a 0 to those in B. The perceptron can only form *linear* boundaries between categories (a straight line in a 2 dimensional space, a flat plane in 3-D, etc.), hence they can only form categories that are *linearly separable* (see Figure 2.1).

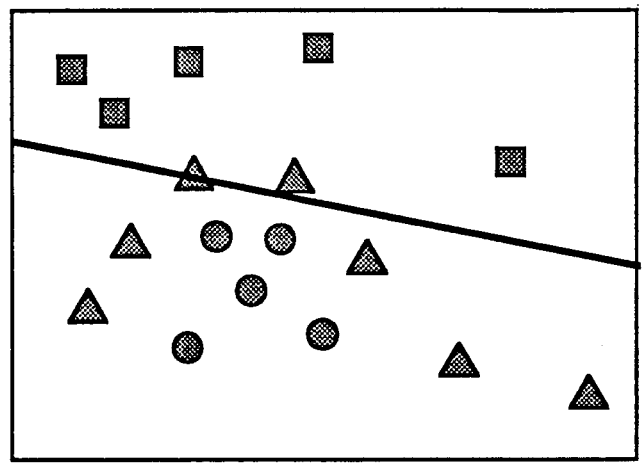


Fig. 2.1. Linear separability is a necessary condition for the success of perceptron learning. The axes represent input activations on the two input lines, each point corresponds to a possible input pattern. The shapes connote category membership. Here, the circle category is separable from the set of squares, but not from the triangles.

The Perceptron Learning Procedure is an error correction procedure applied to a linear threshold unit; that is, unit which responds $r=0$ if $x<\theta$, and $r=1$ if $r\geq\theta$, where x is the weighted sum of the inputs. The weights and the threshold θ are parameters initialized to random values and subsequently optimized by the learning procedure. The algorithm can be conceptualized visually since the weights and thresholds determine a boundary in the pattern space. The error is reduced by selecting data points randomly and checking the correct class against the unit's response. If the point is classified correctly, there is no error and hence no change to the parameters; however, if the unit has misclassified the point, the parameters are changed such that the line moves a fixed distance in a direction towards the point. Thus if the line was sufficiently close to the point, it will cross, and become correctly classified; otherwise, it just moves closer to the correct category (see Figure 2.2). Note that when a particular pattern induces change it never moves from a correct category to an incorrect category, however this fate may befall another pattern in the training set. In spite of this, after many repetitions of the entire training set, the unit is guaranteed to settle in the solution state (assuming the categories were linearly separable).

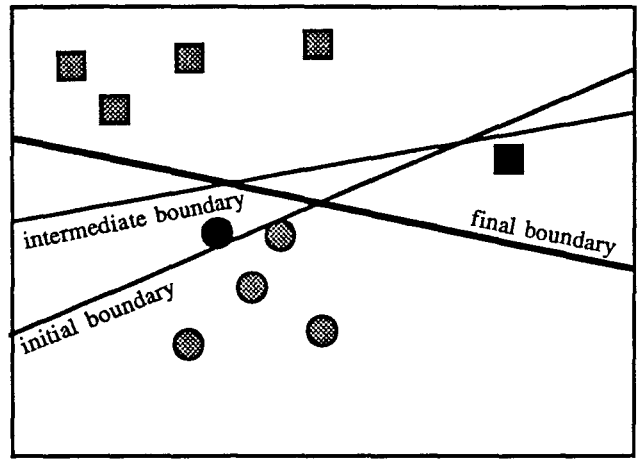


Fig. 2.2. The perceptron learning procedure is visualized at three stages. Initially, two items are misclassified. Presentation of one misclassified item, the dark circle, induces a change in the parameters, such that it is correctly classified (intermediate boundary). The second misclassified item (the dark square) changes the parameters such that all points are correctly classified, so the algorithm stops.

2.2 The Multilayered Perceptron

Unfortunately, linear separability is a very harsh restriction, since it does not include most tasks of interest. Rosenblatt proposed using multiple layers, since such a network can handle problems of arbitrary complexity (see Figure 2.3). All units in the network are linear threshold units (in the figure, all information flow is upward). He showed that there exists a mapping from the input to the hidden layer that will render any classification task linearly separable in the hidden unit space, at least if the original inputs are binary valued. However, the solution may require the introduction of extra parameters (in this case, extra hidden units), and can have an adverse effect on generalization.

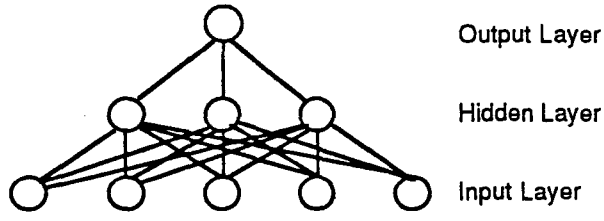
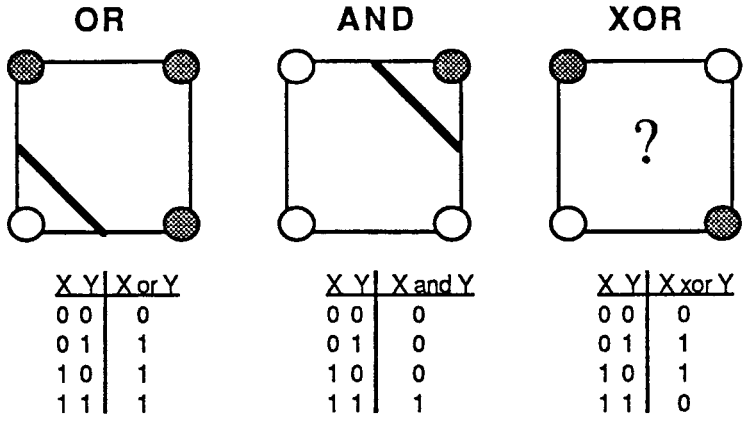


Fig 2.3. A multilayered perceptron consists of at least two layers of weights. So-called "hidden" units lie between the input and output layers.

The effectiveness of the intermediate layer is nicely illustrated by exclusive or (XOR), the simplest boolean task that is not linearly separable (see Figure 2.4). A multilayered perceptron can compute XOR using two hidden units, where one hidden unit computes X OR Y, and the other computes X AND Y (see Figure 2.5).

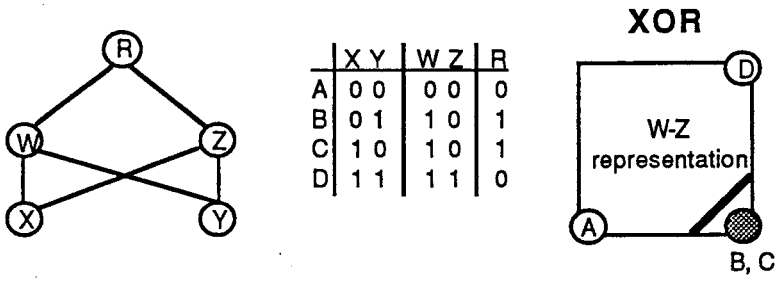


X	Y	X or Y
0	0	0
0	1	1
1	0	1
1	1	1

X	Y	X and Y
0	0	0
0	1	0
1	0	0
1	1	1

X	Y	X xor Y
0	0	0
0	1	1
1	0	1
1	1	0

Fig 2.4. Boolean tasks are shown with standard truth tables and plotted on the X-Y plane. Each combination of X and Y is a corner of the unit square. Gray indicates a truth value of 1, and white corresponds to 0. The diagonal lines indicate where a linear category boundary could be placed.



	X	Y	W	Z	R
A	0	0	0	0	0
B	0	1	1	0	1
C	1	0	1	0	1
D	1	1	1	1	0

Fig 2.5. A network with linear threshold units can compute functions that the individual units cannot compute. The input representation (X-Y) is *not* linearly separable, whereas, the intermediate representation (W-Z) is.

While Rosenblatt was on the right track, he was unable to devise a method by which such a network could be trained by example. The impasse was finally breached with the work of Werbos (1974), and was rediscovered about 12 years later by three research groups working independently (Le Cun, 1986; Parker, 1985; Rumelhart, Hinton, and Williams, 1986). By using units that were not quite threshold units, they were able to derive an error-correction procedure for training multilayered perceptrons, that has generally come to be known as the technique of backward propagation of error, or "backprop".

3 Discussion

Most cognitive models that use neural networks, especially those using backprop, make a loud disclaimer that the units do not correspond to biological neurons, and emphasize that they do *not* mean to assert that backprop is neurally plausible. Neurobiologists agree that these models are a far cry from the real brain. Nevertheless, these models feel much closer to the biological substrate than previous models, and have given successful accounts of many cognitive phenomena, with unprecedented detail.

But certain aspects of neural computation captured by network models provide psychologists with new tools for describing and conceptualizing cognitive phenomena. For example, the following lesson of the multilayer perceptron may give new insight to the nature of biological computation: neural systems compute functions far more complex than is possible for individual neurons, by gradually transforming the representation space, layer by layer, reducing the complexity of the computation to be performed with each step; then, at the penultimate stage, there is a representation that renders the task computable for the individual neurons at the output level.

Studies of neuronal response at the intermediate stages of the systems of animals have shown correspondences with model systems trained by backpropagation (e.g. Zipser and Andersen, 1988). Zipser and Andersen stress that in their simulation of posterior parietal neurons in monkey cortex, they are making claims about the computations performed by the system *after* training, but expressly *not* about the training procedure itself (backprop): "That the back-propagation method appears to discover the same algorithm that is used by the brain in no way implies that back propagation (sic) is actually used by the brain".

Analyses of intermediate representations across several stages may have implications for processes involving manipulation of concepts, such as analogical reasoning, and may shed light on controversies involving competing theories for mental representations, including the symbolic vs. subsymbolic debate in AI, the imagery vs. propositional debate in cognitive neuroscience, and the temporal vs. spatial debate in spatial knowledge acquisition.

Over the last 10 years or so, neural network models have had an increasingly pervasive impact on our understanding of human information processing, and hold the hope (if not the promise) of explaining cognitive processes in a biological (or pseudobiological) framework. One of the most compelling features of this approach is its intrinsic capacity to describe and simulate learning phenomena, which in turn reveal more about the end product of the learning process: the mind.

References

Hebb, D. O. (1949) The Organization of Behavior. New York: Wiley

Hopfield, J. J. (1982) Neural networks and physical systems with emergent collective computational properties. Proceedings of the National Academy of Sciences 79, 2554-2558

Le Cun, Y. (1986) Learning processes in an asymmetric threshold network. In: E. Bienenstock, F. Fogelman Souli, G. Weisbuch (eds.), Disordered systems and biological organization. Berlin: Springer

Parker, D. (1985) Learning Logic, MIT Center for Computational Research in Economics and Management Science Technical Report TR-87. Cambridge MA

Werbos, P. J. (1974) Beyond regression: new tools for prediction and analysis in the behavioral sciences., Ph.D. thesis, Harvard University, Cambridge MA.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986) Learning internal representations by error propagation. In: D. E. Rumelhart, J. L. McClelland (eds.) Parallel distributed processing: Explorations in the microstructure of cognition. Cambridge: MIT Press

Rumelhart, D. E., McClelland J. L. (1986) Parallel distributed processing: Explorations in the microstructure of cognition. Cambridge: MIT Press

Zipser, D. and Andersen, R. A. (1988) A back-propagation programmed network that simulates response properties of a subset of posterior parietal neurons. Nature, 331, 679-684

Laws of visual perception and their consequences for the user interface

Floris L. van Nes
Institute for Perception Research/IPO
PO Box 513
5600 MB Eindhoven
The Netherlands

Abstract

Knowledge of visual perception is needed to display text and graphics in an effective and efficient way. This paper therefore describes the visual processes involved in reading and the effects of typography, spatial layout and text colours on legibility. Some information is given on graphics, as an alternative for text and as the main constituent of graphical user interfaces.

Keywords: visual displays, legibility, typography, font design, layout, text colours, graphics, graphical user interfaces, reading.

1. Introduction

The dominant role of vision in daily life also applies in information technology: the visual display of output from computers and other information processing equipment far outweighs the use of other modalities. The user interface generally comprises a visual display, and perceiving the information presented there is an important part of any task of the user. Good user interfaces therefore must possess good visual displays, i.e. ones that are effective and can be used efficiently, also in the long run. The designer of such effective and efficient visual displays, be it for use by professionals or the general public, therefore needs to know the laws of visual perception. This paper aims to provide a start for the acquisition of that knowledge, as well as guidance for further study, to result in the proper presentation of alphanumerical as well as pictorial material on visual displays.

2. Alphanumeric text

2.1 Reading symbols, words and codes

The majority of the information on visual displays consists of letters, digits, punctuation marks and some special symbols, that all need to be read by the user. The laws that govern the reading process partly have to do with visual factors, partly with linguistic factors and partly with cognitive ones. We will confine ourselves here mainly to the visual factors because they are the ones influenced by the display.

2.2 Reading and visual search

Reading starts with searching. In most reading tasks, the reader does not want to read all symbols that are presented; in fact he or she often wants to read as little as possible, for instance when a dictionary or a telephone directory are consulted. So the reader's eyes hunt for the information of interest - a hunt that may be facilitated or hindered by the way the text is put on the display - by its layout, its contrast and colours and its typography. Headings as are also used in this book, for example, are a very effective means of organizing a text and thus helping the visual search process - provided that the headings stand out from the other text through the use of capitals or bold face, and/or through surrounding a heading with empty space. By such measures, the heading is given a certain degree of conspicuity (Engel, 1980).

2.3 The visual reading process

During normal reading of continuous text the eyes do not move along the lines smoothly, but in a series of rapid movements, the so-called saccades, alternated with fixation pauses in which the eyes are focussed on successive points of the text lines. The saccades in the forward direction are typically 8 4 letter positions long and the fixation pauses last about 250 ms (Roufs and Bouma, 1980). In these pauses the text characters are imaged on the central parts of the retinas of both eyes. Then recognition of the text takes place in the visual reading field, being 10-20 letter positions (Bouma, 1980). Both letter recognition and word contour recognition occur simultaneously, leading to word recognition (Bouwhuis, 1979). Word contour recognition is aided by the presence of ascenders and descenders in lower-case letters, since these lend a characteristic contour to a word - therefore, text made up of mixed lower- and upper case is easier to read than text in upper case (capitals) only. If the overall legibility of a text is low, for instance because the luminance contrast between the characters and their background is low, readers may confuse a word with another one which has the same contour, but one or more different letters (Bouma, 1973). This tendency may be reduced by choosing character configurations that are distinct; for instance a /c/ that has a large opening, to avoid confusion with /o/ or /e/ (Bouma and Van Rens, 1971).

Legibility has been defined by Tinker (1964) and refers to the visual properties of a text. This in distinction to a text's *readability*, which has been defined by Klare (1969) as referring to its linguistic properties, and determining the comprehension of a text after it has been visually recognized. Linguistic properties include stylistic factors such as sentence length, type of vocabulary used, etc.

At the end of a line the eyes have to go to the beginning of the next line, and fixate approximately there. Such eye movements are backward saccades; the ones just described may be called normal backward saccades. Other backward saccades, with a much smaller size, may occur within a text line if something there is not recognized or understood properly, because of a legibility or readability that is too low. Such small backward saccades cannot be called abnormal, but should occur only rarely. The normal, or regular backward saccades have to be directed rather accurately, to prevent a fixation at the beginning of a too low or too high text line. This required accuracy is relatively more important for small angles between backward saccades and the direction of the text lines. This is the reason for the reduced legibility of tightly-packed text, which has short inter-line distances compared with the line lengths.

2.4 Legibility and typography

In the two previous sections, 2.2 and 2.3, several aspects of typography were already mentioned: upper or lower case, different type faces such as bold or italics, and different character configurations i.e. type fonts. In principle there is no difference between the requirements on typography for paper or for visual displays. Indeed high-resolution displays have typographic capabilities comparable to paper and the corresponding possibilities to use different fonts of high legibility, in normal, bold or italics type and of course with both upper and lower case. However, medium- or low-resolution displays have a more limited typographic repertory, with rather coarse characters, typically in a 5 x 7 dot matrix. The design of character sets with optimal legibility within such severe constraints requires an analysis of the relevant character aspects. Such an analysis was done by Bouma and Leopold (1969) leading to requirements for three such aspects:

1. Acceptability. Configurations of characters should be chosen so as to yield a high acceptability, i.e. high degree of correspondence between the dot matrix configuration of, for instance, a letter and the internal image of this letter that people have.
2. Detail identifiability. The details of characters should stand out from their background clearly. This is especially important for inner details, such as the horizontal stroke of the lower case /e/.
3. Individuality, or discriminability. The chosen configurations for similar symbols, such as /c/ and /o/, or /n/ and /h/ should yield a high discriminability of these symbols, especially if they need to be recognized on their own strength, as in codes, without the support of the redundancy of normal language.

One of the challenges of designing these coarse characters is that the requirements for acceptability and individuality may be in conflict with each other. This is shown in Fig. 1 for the upper case characters /M/ and /Y/ from Bouma and Leopold's research (1969). Not too many graphical designers have taken up this challenge; in fact the very requirement of individuality is somewhat in conflict with a basic principle for the design of a particular font as applied by most graphical designers: a certain degree of commonality between the configurations for the letters from a particular alphabet, so as to carry its specificity. Still, in some instances graphical designers already in the early days of matrix characters for displays created those fonts (Crouwel and Dirken, 1973; Unger, 1977), but by and large the field was long left to engineers. Ergonomists then pointed out the necessity to evaluate the legibility of matrix characters, in order to choose the best configurations (Vartabedian, 1973; Huddleston, 1974; Maddox, Burnett and Gutmann, 1977; Snyder and Maddox, 1980).

Indeed visual displays may be used in conditions where legibility is especially critical. The affected legibility factor is luminance contrast: it typically is lower for visual displays than for printed paper, because of reflection on the glass front of the display. Another affected legibility factor is viewing distance: in videotex applications such as Teletext it commonly is twice as large, or more, as the viewing distance recommended for the character size applied (Van Cott and Kinkade, 1972). A special character set was therefore designed for Teletext decoders, and extensively tested as to its discriminability (Van Nes, 1986b). This set, shown in Fig. 2, was designed within a matrix of 12 x 10 elements (horizontal x vertical, including inter-character and

and inter-line gaps). One of the features of this IPO-Normal character set is that the vertical numeral strokes are 1.5 times as wide as the vertical letter strokes, to facilitate the distinction between similar numeral-capital pairs such as /5/ and /S/, /8/ and /B/, and, especially, /0/ and /O/.

2.5 Legibility and layout

2.5.1 Line length and line spacing

As was explained in section 2.3, legibility is reduced if the space between successive lines is small compared to their length, because of the then required small angle, with a small tolerance, between backward saccades and text lines. A minimum value of 0.033 has been recommended for the inter-line space/line length ratio (Bouma, 1980). If this ratio is computed for consecutive text lines in videotex systems such as Teletext, a value of 0.035 is found. In other words, the legibility of a regular videotex page is only moderate - as indeed may be observed in practice. Possible measures to increase this legibility are an increase of the inter-line spaces, which may be achieved by only filling alternate videotex lines with characters, or a reduction of the actual text line length, for instance by only putting 20 characters on one line, instead of the maximum 40. Both measures will of course halve the text-carrying capacity of the page; but if two narrow columns of text are employed, as in a newspaper, the original capacity is almost reached again (Van Nes, 1986a).

2.5.2 Spatial text grouping

Even with long text lines at a short distance legibility may be influenced positively by the editor of the text: through spatial text grouping by the insertion of empty lines, as in Fig. 3, where the whole text is split up in two paragraphs. This figure also shows that lines which are only partially filled, such as the fourth one, tend to improve legibility, because then it is easier for the eyes, i.e. the direction of gaze, to find the start of the next text line, in this case the fifth one.

Spatial grouping is especially important in tables, where it may facilitate search for the desired items. Tables are, therefore, generally organized in vertical columns, separated by quite a bit of empty space, or by thin vertical lines. But a horizontal organization also helps to guide the eye. Good tables should have an empty line between groups of about five filled lines. Figs. 4 and 5 also may be called tables, with a horizontal organization. The information that is grouped here refers to programmes broadcasted by three radio stations, Hilversum 1, Hilversum 2 and Hilversum 3. The font used in Fig. 4 is the old-fashioned 6 x 10 dot matrix Teletext font, whereas the font used in Fig. 5 is the current 'IPO-Normal' 12 x 10 dot matrix font (see section 2.4).

2.6 Legibility and colour

The application of colour in the display of text and graphics has increased very much over the last 20 years. This is primarily caused by progress in colour display technology, but also by the inherent attraction of coloured images in general. Unfortunately, the growth in numbers of colour displays has not been paralleled by a

similar growth of insight in the effects on perception, both of text and graphics (De Weert, 1988). Yet the rules for an advantageous use of colour are fairly simple.

2.6.1 Recognition of coloured text

First of all a coloured text should be seen at all, i.e. be recognized. This recognition mainly depends on the luminous contrast between text characters and background, both for bright and dark backgrounds. Colour contrast plays only a small role (Bruce and Foster, 1982). This is of importance in systems such as videotex, that do not compensate for the luminous efficiency differences between the red, green and blue phosphors. It means that the eight available 'colours': white, yellow, cyan, green, magenta, red, blue and black differ in luminosity and brightness; from high to low in this order. On a dark or 'black' background the brighter colours should be used for the characters whereas on a bright background the darker colours should be used, to provide sufficient luminance contrast.

2.6.2 Accentuation by colour difference

If some characters from a text have another colour than the surrounding characters they will be conspicuous (Engel, 1980) and thus may be perceived as being accentuated. On low-resolution displays that do not permit the use of different type faces such as bold or italics, colour is one of the few remaining means to accentuate a text part. However, colour is so powerful in this respect that an overemphasis easily occurs (Van Nes, 1991). A moderate emphasis has been lent to the headings 'Hilversum 1, 2 and 3' in Figs. 4 and 5 by their cyan colour, among otherwise differently coloured characters.

2.6.3 Coding text by colours

Parts of a text may be coded by giving them a specific colour. Such a code then attaches a particular meaning to those text parts, in addition to their normal semantic meaning. As with all codes, this code must be known to be useful for the reader. Its usefulness is diminished if it is not employed consistently in the whole text concerned. A (somewhat inconsistent) application of such a code in Teletext systems can be seen in The Netherlands and, to a limited extent, in Belgium: cyan, or light blue text in a programme guide refers to programmes that are at the moment *not* available - so it makes no sense to try and select them.

2.6.4 Associating text parts by colour grouping

Text parts or figure fragments with the same colour are perceptually grouped, i.e. seen as belonging together. This association mechanism operates more or less autonomously - at least as long as not too many different colours are present in the text or figure, typically three or four, according to Reynolds (1979). The association, or grouping effect has been attributed to the formation of a 'Gestalt' by equally coloured image parts. These Gestalts are assumed not to be formed or to break down, if too many colours are present in the image (Cahill and Carter, 1976). An inverse effect may be observed fairly often in practice: parts of a text (or figure) that have a different colour are difficult to be seen as related. This may actually hamper the understanding of a

sentence that is displayed in two colours because it is basically perceived as two unrelated parts (Van Nes, 1991).

3. Graphics

3.1 The choice between pictures and words

Whether to use a picture or text to render visually displayed information depends on a variety of issues, from unspecific but very real factors such as avoiding an impression of monotony or dullness to specific ones, for instance the well-documented superiority of a graph over a table for interpreting the relationship between two sets of numbers (Oborne, 1987). Tullis (1988) mentions a number of instances that support the well-known Chinese proverb: "A picture is worth a thousand words". However, the choice certainly also depends on the context of the reader's task. A recent study of how engineers use either graphical representations or text for electronics suggests that either graphics or text may be preferred, depending on the accessibility of information needed while the engineers perform their tasks (Petre and Green, 1990).

3.2 Graphical User Interfaces

Just as is the case with text, nowadays electronic displays can represent graphics (almost) as well as printed paper. In fact such displays, together with the underlying computing power, may give an extra dimension to graphics, for example animation (Tiritoglu and Juola, elsewhere in this volume). In general, the advent and success of Graphical User Interfaces has spurred an interactive use of graphics and therefore, created an increased importance of the perceptual laws governing the recognition and interpretation of graphical images such as 'icons', the small representations of 'windows' when they are in their closed state. In their open or visible state 'windows' are bordered screen areas with a specific content. A considerable development in the use of graphics may still be expected, possibly through a joint effort of various professionals such as cartographers, graphic designers, visual perception experts and cognitive scientists. Such a development is needed because presently the abundant use of graphics is known to sometimes distract the user (Billingsley, 1988, p. 421). One should not forget that the graphical ingenuity as described by Verplank (1988), having resulted in the generally praised user interface of the Xerox Star (Johnson et al., 1989) may be contrasted by lists of common errors in graphical design as compiled by Wainer (1980) and by the many cases of 'deceptive graphics' collected by Tufte (1983, p. 53). Although many practical guidelines exist for the design of good graphics (Tufte, 1983, pp. 91-190) there as yet is no graphical counterpart of the theoretical foundation for alphanumeric legibility research and knowledge (Twyman, 1979).

4. Conclusion

For the best visual displays electronic text may now have nearly the same quality as printed text. Consequently, the legibility of electronic text will soon be the same as that of printed text. This does not imply, however, that reading *multipage* printed text

such as books can be done with equal ease and speed in the case of multipage electronic text (Wright, 1989). On the other hand electronic books can be equipped with indexing systems that are superior to the passive indexes from books; and indeed information retrieval from such 'electronic books' has been shown to be superior to information retrieval from a printed book of similar content (Egan et al., 1989; Leventhal et al., 1993).

5. References

Billingsley, P.A. (1988). Taking panes: issues in the design of windowing systems. In: Handbook of Human-Computer Interaction (M. Helander, editor), pp. 413-436. Amsterdam: Elsevier Science Publishers BV - North-Holland.

Bouma, H. (1973). Visual interference in the parafoveal recognition of initial and final letters of words. Vision Research, 13, pp. 767-782.

Bouma, H. (1980). Visual reading processes and the quality of text displays. Ergonomic Aspects of Visual Display Terminals (E. Grandjean and E. Vigliani, editors), pp. 101-114. London: Taylor & Francis Ltd.

Bouma, H. and Leopold, F.F. (1969). A set of matrix characters in a special 7 x 8 array. IPO Annual Progress Report, 4, pp. 115-119.

Bouma, H. and Rens, A.L.M. van (1971). Completion of an alphanumeric matrix display with lower-case letters. IPO Annual Progress Report, 6, pp. 91-94.

Bouwhuis, D.G. (1979). Visual Recognition of Words. Catholic University of Nijmegen. Ph.D. dissertation.

Bruce, M. and Foster, J.J. (1982). The visibility of colored characters on colored backgrounds in viewdata displays. Visible Language, 16, pp. 382-390.

Cahill, M.C. and Carter, Jr, R.C. (1976). Color code size for searching displays of different density. Human Factors, 18, pp. 273-280.

Cott, H.P. Van and Kinkade, R.G. (editors) (1972). Human Engineering Guide to Equipment Design. Revised ed. p. 107. Washington DC: American Institutes for Research.

Crouwel, W.H. and Dirken, J.M. (1973). Alphanumeric symbols for mosaic printers and display tubes. Icographic, 6, pp. 12-14.

Egan, D.E., Remde, J.R., Landauer, T.K., Lochbaum, C.C. and Gomez, L.M. (1989). Behavioral evaluation and analysis of a hypertext browser. In: Proceedings CHI '89 Human Factors in Computing Systems, pp. 205-210. New York: ACM, 1989.

Engel, F.L. (1980). Information selection from visual display units. In: Ergonomic Aspects of Visual Display Terminals (E. Grandjean and E. Vigliani, editors), pp. 121-125. London: Taylor & Francis Ltd.

Huddleston, H.F. (1974). A comparison of two 7 x 9 matrix alphanumeric designs for TV displays. Applied Ergonomics, 5 (2), pp. 81-83.

Johnson, J., Roberts, T.L., Verplank, W., Smith, D.C., Irby, C.H., Beard, M. and Mackey, K. (1989). The Xerox Star: A retrospective. Computer, 22 (9), pp. 11-29.

Klare, G.R. (1969). The Measurement of Readability, pp. 1-2. Ames, Iowa: The Iowa State University Press.

Leventhal, L.M., Teasley, B.M., Instone, K., Rohlman, D.S. and Farhat, J. (1993). Sleuthing in HyperHolmes (TM): an evaluation of using hypertext vs. a book to answer questions. Behaviour and Information Technology, 12 (3), pp. 149-164.

Maddox, M.E., Burnett, J.T. and Gutmann, J.G. (1977). Font comparisons for 5 x 7 dot matrix characters. Human Factors, 19, pp. 89-93.

Nes, F.L. van (1986a). Space, colour and typography on visual display terminals. Behaviour and Information Technology, 5 (2), pp. 99-118.

Nes, F.L. van (1986b). A new Teletext character set with enhanced legibility. IEEE Trans. Electron Devices (ED-33), 8, pp. 1222-1225.

Nes, F.L. van (1991). Visual Ergonomics of Displays. Chapter 6 of The Man-Machine Interface (J.A.J. Roufs, editor), Vol. 15 of Vision and Visual Dysfunction. London: The MacMillan Press Ltd., 1991.

Oborne, D.J. (1987). Man-machine communication: words and symbols. In: Ergonomics at Work, 2nd ed., p. 32. Chichester: John Wiley & Sons.

Petre, M. and Green, T.R.G. (1990). Where to draw the line with text: Some claims by logic designers about graphics in notation. Proc. of Human-Computer Interaction - INTERACT '90 (D. Diaper, D. Gilmore, G. Cockton and B. Shackel, editors), pp. 463-468. Amsterdam: Elsevier Science Publishers BV - North-Holland.

Reynolds, L. (1979). Teletext and viewdata - a new challenge for the designer. Information Design Journal, 1, pp. 2-14.

Roufs, J.A.J. and Bouma, H. (1980). Towards linking perception research and image quality. Proc. Soc. Information Display, 21 (3), pp. 247-270.

Snyder, H.L. and Maddox, M.E. (1980). On the image quality of dot-matrix displays. Proc. Soc. Information Display, 21 (1), pp. 3-7.

Tinker, M.A. (1964). Legibility of Print. Ames, Iowa: The Iowa State University Press.

Tiritoglu, A. and Juola, J.F. (1993). Animated icons promote learning of their functions. This volume.

Tufte, E.R. (1983). The Visual Display of Quantitative Information. Cheshire, Connecticut: Graphics Press.

Tullis, T.S. (1988). Screen design. In: Handbook of Human-Computer Interaction (M. Helander, editor), pp. 377-411. Amsterdam: Elsevier Science Publishers BV - North-Holland.

Twyman, M. (1979). A scheme for the study of graphic language. In: Processing of Visible Language. Vol. 1. (P.A. Kolers, M.E. Wrolstad and H. Bouma, editors), pp. 117-150. New York: Plenum Press.

Unger, G. (1977). Telefoongidsen: nieuwe typografie. Compres, 2 (8).

Vartabedian, A.G. (1973). Developing a graphic set for cathode ray tube display using a 7 x 9 dot pattern. Applied Ergonomics, 4 (1), pp. 11-16.

Verplank, W.L. (1988). Graphic challenges in designing object-oriented user interfaces. In: Handbook of Human-Computer Interaction (M. Helander, editor), pp. 365-376. Amsterdam: Elsevier Science Publishers BV - North-Holland.

Wainer, H. (1980). Making newspaper graphs fit to print. In: Processing of Visible Language, Vol. 2. (P.A. Kolers, M.E. Wrolstad and H. Bouma, editors), pp. 125-142. New York: Plenum Press.

Weert, C.C.M. de (1988). The use of colour in visual displays. Human-Computer Interaction: Psychonomic Aspects (G.C. van der Veer and G. Mulder, editors), pp. 26-40. Berlin: Springer-Verlag.

Wright, P. (1989). The need for theories of NOT reading: some psychological aspects of the human-computer interface. In: Working Models of Human Perception (B.A.G. Elsendoorn and H. Bouma, editors), pp. 319-340. London: Academic Press.

Captions

Fig. 1. Two examples of a conflict between acceptability and identifiability (from Bouma and Leopold, 1969).

Fig. 2. The basic set of IPO-Normal 12 x 10 dot matrix characters (copyrighted). The complete IPO-Normal set is protected under the rules of the International Design Registration effected under the Geneva Protocol of 1975.

Fig. 3. The division of this text in two paragraphs as well as the only partial filling of the fourth text line improve legibility.

Fig. 4. (in colour) A table with a horizontal organization; the empty lines above 'Hilversum 2' and 'Hilversum 3' clearly distinguish the programmes of the radio stations Hilversum 1, Hilversum 2 and Hilversum 3. The font used here is the old-fashioned 6 x 10 dot matrix Teletext font. In order to distinguish the digit zero from the capital O, a diamond shape was chosen.

Fig. 5. (in colour) The same table as shown in Fig. 4, but now with the current IPO-Normal 12 x 10 dot matrix font. In this font all numerals are bold compared to the letters, to facilitate the distinction between, e.g., the digit zero and the capital O.

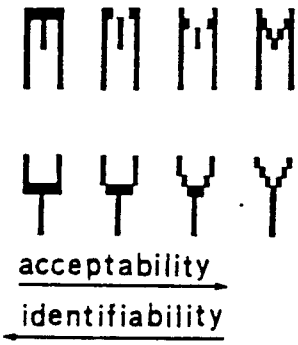


Fig. 1

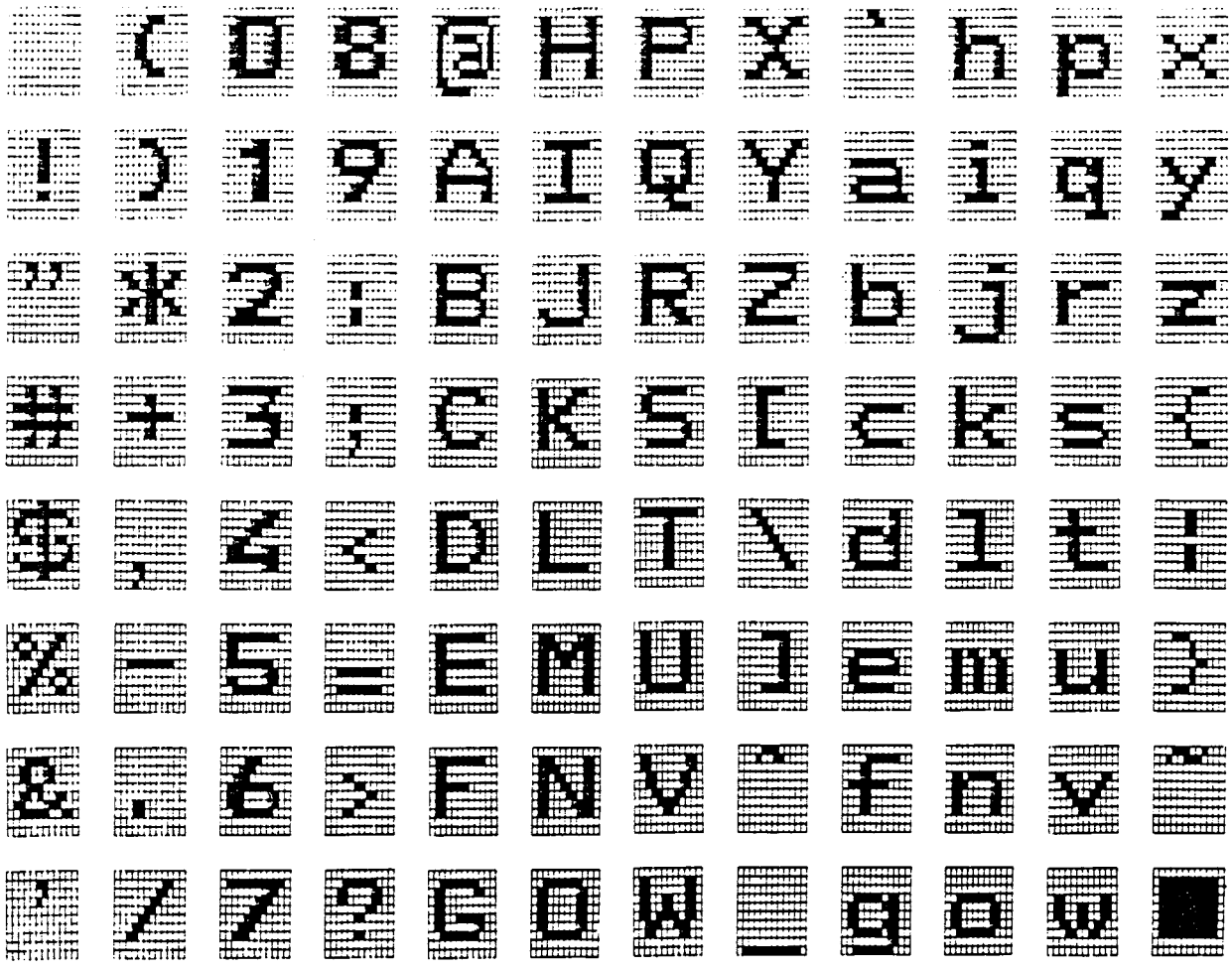


Fig. 2

Fig. 4 →

actualiteiten OP RADIO 16/11 2

Hilversum 1

- "Aktua" (TROS) om 08.06, 13.06 en 18.06 uur.
- "Met het oog op morgen" (NOS) om 23.02 uur.

Hilversum 2

- "Hier en nu" (NCRV) om 07.36, 08.33 12.36, 17.36 en 18.10 uur.

Hilversum 3

- "AVRO's Radiojournaal" om 07.30, 08.30, 12,30 en 17.30 uur.

(slot)

↓ Fig. 3

uit de dagbladen 1

NRC Handelsblad constateert dat de regeringsverklaring wemelt van de vaagheden waarmee schijnoplossingen vaak gepaard gaan.
De krant twijfelt en geeft de regering nauwlijks het voordeel van de twijfel.

De inkomingsmatiging, zegt de NRC, is tot hoeksteen van het beleid uitgeroepen, zonder dat duidelijk wordt gemaakt hoe matiging moet worden bereikt.
De krant bestrijdt de 'foutieve redenering' als zou de lastendruk niet stijgen doordat de belastingverhoging

↓ Fig.

actualiteiten OP RADIO 16/11 2

Hilversum 1

- "Aktua" (TROS) om 08.06, 13.06 en 18.06 uur.
- "Met het oog op morgen" (NOS) om 23.02 uur.

Hilversum 2

- "Hier en nu" (NCRV) om 07.36, 08.33 12.36, 17.36 en 18.10 uur.

Hilversum 3

- "AVRO's Radiojournaal" om 07.30, 08.30, 12,30 en 17.30 uur.

Basics of Man-Machine Communication for the Design of Educational Systems

NATO Advanced Study Institute

August 16-26, 1993

Eindhoven, The Netherlands
Organized by IPO

Organizing Committee

Maddy D. Brouwer-Janse, ASI - Director,
IPO - Institute for Perception Research, Eindhoven, NL

Dominique Béroule, LIMSI-CNRS, Orsay, F
Robbert-Jan Beun, IPO, Eindhoven, NL
Tom Bösser, WWU, Münster, BRD

Editors

Maddy D. Brouwer-Janse, IPO, Eindhoven, NL
Ilse van Kuijck-Aerts, IPO, Eindhoven, NL

Proceedings

Volume II

IPO

EINDHOVEN

Contents

VOLUME 1

Contents

VOLUME 2

Interactive Learning and Natural Language Systems

Offereins, M.

University of Twente
Dep. of Computer Science
P.O. Box 217
7500 AE Enschede
The Netherlands
e-mail: grietje@cs.utwente.nl

Abstract.

In this paper Interactive Learning is presented as a research application to apply and experiment with Natural Language processing.
Computer-Assisted Instruction (CAI) systems are useful and applied successfully in many domains now. CAI systems provide instruction in a restricted knowledge domain mostly without using a sophisticated Natural Language (NL) interface. The CAI interface only supports the use of predefined natural language utterances that are derived directly from the knowledge domain.
The biggest problem with applying natural language processing tools to CAI is the lack of a direct link between the architecture of the NL interface and the architecture of a CAI system. This paper is a proposal for solving it.
There are two approaches presented. One approach focusses on defining a NL interface on top of a CAI system. The other approach focusses on Interactive Language Learning.
The first approach stresses that applying Natural Language dialogues to CAI impoverishes the way instruction is provided by the system, unless the system is based on discourse planning.
The second approach stresses that Natural Language processing benefits CAI systems for Language Learning. It offers a learning environment for different language skills, based on a variety of natural language processing tools.

Keywords: Natural Language Communication, Human Learning, Interactive Learning, CAI.

1. Introduction.

People learn by communicating with each other in a natural language. In Interactive Learning computers are applied for human learning in a particular knowledge domain. It is our intention to find out in what way Natural Language is used in Interactive Learning.
Assuming that the user is the learner and the system knows everything about the knowledge domain, there are two extremes in Interactive Learning. One extreme is that the system is in control and provides instruction to the user. The other extreme is that the user is in control and decides on what he wants to learn. In other words, Interactive Learning is either system-oriented or user-oriented. In the user-oriented

approach, the user should be offered different ways to access the learning material. Applying Natural Language is one of them.
Most Computer-Assisted Instruction (CAI) systems have been developed by structuring and implementing the knowledge domain. The interface environment presents pieces of knowledge in the way they are structured and accessible in the system. These system-oriented CAI systems do not rely on plain natural language processing, but produce prepared pieces of texts. The use of natural language does not support the structuring and the accessibility of the knowledge that is taught by the system. It is only needed to explain, give hints, help and feedback to the learner. It will cause a lot of overhead if Natural Language understanding is applied to CAI, but there are instruction styles that are based on discourse planning.
In Computer-Assisted Language Learning (CALL) the same problems occur, but the difference is that the knowledge domain itself is about Natural Language. Applying Natural Language dialogues in these systems is supported by knowledge on vocabulary, syntax and grammar of a language. An integration of both applications is valuable. It provides spin-offs for Natural Language interaction in general and for Interactive Learning in your own natural language.

After this introduction four sections follow. The first section looks into Interactive Learning and Instruction and show what we mean by a CAI system. The second section discusses Natural Language interfaces for CAI. The third section looks at using Natural Language as the knowledge domain for Interactive Learning in CALL. The final section gives the conclusions.

2. Interactive Learning and Instruction.

Interactive Learning is applied in different knowledge domains. In Computer-Assisted Instruction (CAI) systems the learning domain is bounded and controlled by the system. Therefore a CAI system is an appropriate application for Natural Language communication. However, the system is well-designed and specially made to support a particular way of instruction of the learning material. An analysis of natural language dialogues for instruction is required to find out if there is a need to develop Natural Language interfaces for CAI.
First we present a general architecture of a CAI system from the knowledge perspective. Then we give interface criteria for Interactive Learning.

2.1. General architecture of CAI.

A general architecture of a CAI system is depicted in the figure below (Figure 1) and consists of knowledge components that are interrelated in various ways.

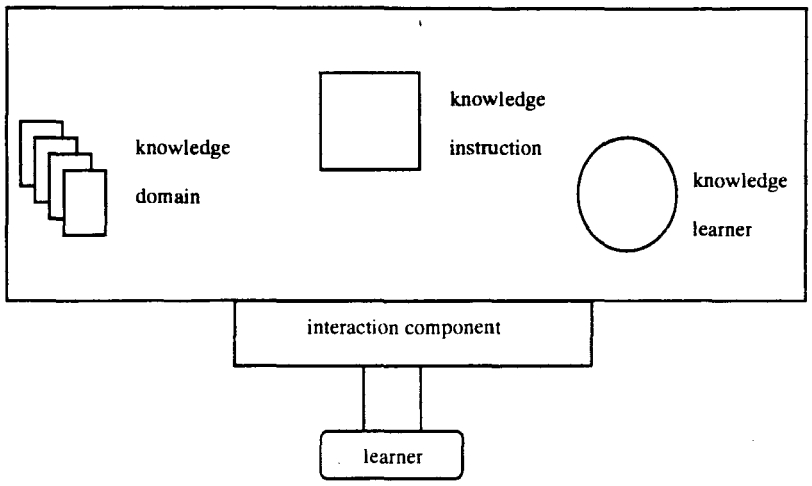


Figure 1. *The general knowledge components of a CAI system*

Different knowledge representation formalisms are used for each knowledge component at different knowledge levels. It is not our intention to describe them here in detail. The instruction structures the knowledge domain and controls the interaction with the learner by making use of knowledge on the learner's performance in learning.
As an example, we look at a CAI system for a problem solving domain, such as algebra or geometry. Exercises will be presented to the user. The solution attempts of the user are traced and evaluated. Feedback is provided on errors and the next exercise is selected based on the performance of the learner. The table (Table 1) gives an overview of knowledge components that are involved.

Knowledge Components	Problem Solving
Knowledge Bases	exercises, user model, solutions, feedback, bug catalogue
Instruction Tasks	exercise selection, diagnosing learner's solution, selecting appropriate feedback, evaluating learner's performance
Interface Environment	
Knowledge Sources	learner's solution, feedback, questions, menu selection, mouse clicks, keyboard strokes
Tools	code window, text editor, programming language environment

Table 1. *Knowledge Components of a CAI system in Problem Solving*

2.2. Discourse planning and instruction.

The instruction dialogues in CAI are hardly based on the use of natural language processing techniques. Instead, prepared text fragments are used and filled in with context information. The planning and design of instructional dialogues is done merely to support the design of CAI systems (Woolf, 1988).

Dialogues are of use in cooperative problem solving and coaching . The CAI interface's design is more user-oriented. Dialogues are based on analysis of human discourse in performing learning tasks. Analysis of dialogues in cooperative problem solving shows that information is transferred on a content and a communication level (Erkens & Andriessen, 1993). The dialogues are coded with the help of a Verbal Observation System (VOS) based on clue-words in the utterances. The transcriptions include respectively a dialogue act, a propositional content and an illocution. The results of the analysis are used for the development of a Dialogue Monitor for an Intelligent Cooperative Educational System. For the factual interaction with the system a menu-based natural language interface has been constructed (Erkens, Giezeman & Andriessen, 1993). In working with the interface, the learner makes, in reality, a proposition in the internally used VOS-representation of the program.
Research on didactic discourse in Intelligent Help Systems (Winkels, Breuker & Sandberg, 1988; Winkels, 1992) resulted in the development of a generic Coach based on a Discourse Planner. The discourse is meant for support on the understanding as well as on the use of a computer application environment. The learning principles are specified in the form of relations between concepts in the application domain.

3. Natural Language Interfaces to CAI.

Natural Language interfaces are either word or menu-based, sentence or question-answering based, text or dialogue based. They stress a particular instruction style. An instruction style is viewed here as a possible combination of methods and techniques that are used for guiding the user in learning. For example, the amount of feedback that is provided in a certain measure of detail, or the freedom the user is allowed to have in elaborating an exercise without having the system interrupt or control the sequence of steps that are taken. An interface is part of the instruction style. There are criteria for the design of an interface for instruction. Different natural language interfaces

3.1. Interface criteria.

A well-designed interface for CAI enhances the instruction. The following three aspects characterize the style of instruction of a CAI system:

- Communication Means: The interface supports one or more communication means, such as a language for natural language dialogues, a window for system comments, menus, screen graphics, or a workspace for solving exercises.
- Control: The interface puts an amount of control on the user. It shows the user what he is allowed to do. It narrows down the possible user's actions. This is of help for monitoring and identifying what the user's actions are. These actions are evaluated in the instruction process.
- Support: The interface provides an amount of support to the user. A restricted set of communication means, windows with different functionality and menus support the user in following a specific path in learning. A special window with on-line help messages or hints will keep the user activated.

Applying Natural Language dialogues as one of the communication means requires the design of a specific set of language understanding and generation tools.
For the design of Natural Language processing tools for instruction, the following linguistic criteria apply (Burton & Brown, In: Wenger, pp. 56, 1987):

- Efficiency: The learner should not have to wait for the parser to complete its task.
- Habitability: Within its domain, the system should accommodate the range of ways in which learners are likely to express the same ideas.
- Robustness with ambiguities: The system should not expect each sentence to be complete and unambiguous.
- Self-tutoring: The interface should make suggestions and handle unacceptable inputs in such a way as to clarify the scope of possible interactions.

3.2. Instructional interfaces.

In the following Natural Language interfaces to CAI are discussed. They support different instruction and learning styles. Plain dialogues do not benefit a well-designed CAI system. They require extra overhead for communication, at the expense of the instruction in the knowledge domain.

Menu-based Natural Language System.
A Menu-based Natural Language System supports the user in showing the system's boundaries. The way the menu-system is set up narrows down the scope of ambiguity and misinterpretation in language use for both the user and the system. The expressive power of natural language is combined with the ease of use of menus (Shapiro, 1987; Miller, 1988; Erkens, Giezeman & Andriessen, 1993).
For example, in 'Bridge', an ITS for teaching programming, a problem solving task starts out with menu-entries of informal language phrases to support the learner in defining the right plans to solve the problem (Bonar, 1991). The selection of menu-entries also triggers internal knowledge-bases and processes that are attached to the menu-entries and that are relevant for supporting the next phases in the problem solving task.

Hypertext Based System.
A Hypertext Based System provides additional information on items that are highlighted in the text. It is up to the user to select an item to learn what it is about. Each item that can be selected is the top of a stack of one or more hypertexts.
HyperCard is a more elaborated version of this learning approach. The eL system (O'Brien, 1992), for example, has an interface built with HyperCard. The idea is that hypertext cards are used like 'rooms' in a game-playing scenario, where each room represents a context containing the semantic scope of the language used.

Question-Answering System.
A Question-Answering System gives the user complete control in learning. The system has to deal with the user's typing and spelling errors. The domain is restricted to the context of a fixed database of knowledge. The results of parsing a question is mapped to a database query. The answers that are given are based on facts that are explicitly present in the database. It is a problem to include more information in the

answer to provide more helpful reponses to the user. This requires the use of models of the user's intentions and knowledge (Allen, Chapter 16, 1987).

Dialogue System.
A Dialogue System imposes no control on the user. The instructional capabilities of interaction in plain natural language are poor. The user can type any answer or question and leave it up to the system to proceed the dialogue in a way that is making sense. The limits of both the linguistic and the conceptual coverage of the system are difficult for users to infer.
The system has to deal with the user's typing and spelling errors. There is a huge overhead of language processing techniques. Analysis techniques for the processing of natural language dialogues are therefore often word spotting algorithms that are based on the recognition of certain keywords.
For example, the Albert system (Oberem, 1991) is an intelligent physics CAI system designed to provide instruction on one-dimensional kinematics problems. The student can type in the problem description in English. The system conducts a dialogue in English to help the student in analysing the problem. The communication is based on the recognition of keywords in the student's responses to the system's inquiries.

4. Computer-Assisted Language Learning.

Computer-Assisted Language Learning (CALL) is done at a word, sentence or text level. Learning tasks are about spelling and syntax, grammar, text comprehension and composition. These learning tasks can be applied for learning in a native language, but also for learning a foreign language. The variety of learning tasks result in different CALL systems. The integration of Interactive Learning and Language is complete when these learning tasks are all supported by a platform of natural language knowledge and processing tools.

4.1. Language tasks and tools.

Learning a natural language at a word, sentence or text level requires learning tools that support the generation of exercises at these levels.
At the word level, learning tasks focus on a word vocabulary and the spelling of words. The development of dictionaries and vocabularies is important. The learning tasks are produced, for example, by sentence generators.
At the sentence level, learning tasks focus on spelling and grammar. Learning task are, for example, the construction of a complete sentence, the translation of a sentence, the transformation of a sentence from the active into the passive form or the construction of the right verb form in a sentence. The structure of a sentence is of importance. Knowledge about the syntax and grammar is required and can be delivered by several grammar formalisms.
At the text level, learning tasks focus on text comprehension and composition. Knowledge on text writing style conventions is required. However, using a text as a context for exercises at the word or sentence level is also possible.
For example, a text can be explored by a concordancer (Johns, pp. 9-33, In: Bongaerts, e.a., 1988). A concordancer is a computer program that is able to search rapidly through large quantities of text for a target item (morpheme, word or phrase) and print out all the examples it finds with the contexts in which they appear.

Learning tasks at the text level are provided with word or text processors that include spelling, grammar and even style checkers.
For example, the smart word processor of (Pijls & Sandberg, 1989, pp. 28-37; Vosse, 1989) is an integrated package with a Dutch lexicon, a grammar, a spelling detection and correction module, a sentence analyser and a user interface. The lexicon consists of about ten thousand different lemma's of three thousand of frequently used words.
Learning tasks on text writing are provided with word or text processors that support the planning of the writing process (Geest, 1986).

4.2. Interactive language learning systems.

In the following examples are given of CALL systems for different language skills. Learning spelling and grammar in the native language is presented first. In foreign language learning translation is applicable as a learning task. In translation knowledge is used of both the native and the foreign language to judge the performance of the learner.

Learning Spelling and Grammar.
A CALL system in German (Schwind, 1987) consists of a grammar knowledge base, an exercise execution module, a natural language and graphic interface and an error explication module. The grammar knowledge base allows different access modes to respectively analyse sentences, produce sentences, analyse and explain errors and answer student's queries. The interaction is done in either French or German. Exercises are given in dialogue form in which the student may ask questions in the context of the exercise, that is about the system's explanations, the exercise or the properties of the German language in general.
A CALL system in Dutch (Pijls, Daelemans & Kempen, 1987) consists of a linguistic expert system, a didactic module including a subcomponent for diagnosing the learner's knowledge, a bug catalogue, an instruction module, an exercise generator and a student interface with a powerful graphical editor for graphical manipulation and animation of syntactic trees.
The direct manipulation of syntactic tree structures is possible with a tree editor. Different exercises with the tree structure are possible, like transforming the structure of a sentence into a new sentence, making incomplete tree structures complete or building a tree based on the words of a given sentence.
For the conjugation of Dutch verbs with the so called TDTDT program, a decision tree is used. The solution of a specific conjugation problem is represented as a walk through the decision tree.

Foreign Language Learning.
A CAI system for learning non-native speakers English is the system VP^2 (Schuster & Finin, 1986). VP^2 focusses on the acquisition of English constructions formed from a verb plus particle or verb plus prepositional phrase. The exercises comprise the translation of Spanish sentences in English.
This system is of interest as its user modelling is based on a theory about the learning of a new language. According to this theory, learning a new language is done by comparing and contrasting language constructs in the new language with those in the native language.
Therefore, it is advocated that a grammar of the native language is used as a user model for teaching a foreign language (Schuster, 1985). It is of benefit for both the

teaching and the diagnostic strategy, as the learning can be based on learning in analogy or in contrast with native language aspects.
The use of natural language products for foreign language learning may lead to a language independent CALL system (Yazdani, 1988; Yazdani, 1989; Yazdani, 1991). The French Grammar Analyser (FGA) (Barchan, Woodmansee & Yazdani, 1986) is a program in Prolog that presents English sentences for translation in which specific points of grammar are incorporated to test the learner's performance. The grammar rules of French, the parsing techniques and the error reporting modules are kept seperate and distinct from another to support an "open-ended" approach to developing in the CALL environment. The FGA system was followed by the development of the Language Independent Grammatical Error Reporter (LINGER). This system has been used for the design of the eL system (O'Brien, 1992), an automatic syntactic error corrector, intended for use in English learning.
A CALL system for Spanish, CALEB (Cunningham, Iberall and Woolf, 1986; Woolf & Cunningham, 1987, pp. 46-48), uses a production rule based architecture for teaching Spanish. The emphasis is put on the communication between the CAI system and the learner. The system comprises a rich interactive environment, which is text and graphics based. Word-oriented responses are typed at the keyboard and action-oriented responses are performed with the mouse and pictured objects.
CALEB's knowledge of Spanish comprises a limited list of topics and pieces of language, like a phoneme, syllable, word or phrase. The student manipulates these pieces to construct meaningful sentences. CALEB is based on predefined exercises and answers.
Another CALL system for Spanish, the Computer-Assisted Language Learning Environment (CALLE) (Rypa, 1992) is based on linguistic knowledge of syntax and grammar. The dialogue-based system operates in a window environment in which a target text to be translated can be queried to yield linguistic information relevant to the learner. A major goal is to promote inquiry into the patterns of the target language. Natural language processing provides data to diagnose the learner's input.

5. Conclusions.

Natural Language is part of a user-oriented design of CAI interfaces. The dialogues are based on analysis of human discourse in learning tasks. Applying Natural Language does not enhance the instruction of a CAI system and causes a lot of overhead of knowledge and processing tools for the communication, unless the instruction is on Language Learning. In that case knowledge and processing tools on natural language are used for a variety of learning tasks and support natural language dialogue in language learning.

References.

Allen, J.F. (1987). Natural Language Understanding. Menlo Park, CA: Benjamin/Cummings.

Barchan, J., Woodmansee, B., & Yazdani, M. (1986). A Prolog-based Tool for French Grammar Analysis. Instructional Science 15, pp 21 - 48. Dordrecht, the Netherlands: Martinus Nijhoff.

Bonar, J.G. (1991). Interface Architectures for Intelligent Tutoring Systems. In H. Burns, J.W. Parlett & C. Luckhardt Redfield (Eds.): Intelligent Tutoring Systems: Evolutions in Design. Chapter 3, pp. 35 - 67. Hillsdale, NJ: Lawrence Erlbaum.

Bongaerts, T., Haan, P.de, Lobbe, S., & Wekker, H. (Eds.) (1988). Computer Applications in Language Learning. Dordrecht, the Netherlands: Foris.

Cunningham, P., Iberall, T., & Woolf, B. (1986). CALEB: An Intelligent Second Language Tutor. In IEEE Conference on Systems, Man, & Cybernetics, pp. 1210 - 1215.

Erkens, G., & Andriessen, J. (1993). Cooperation in Problem Solving and Educational Computer Programs. To be published in Tennysson, R. (Ed.): Computers in Human Behavior, Summer issue.

Erkens, G., Giezeman, M., & Andriessen, J. (1993). Menu-gestuurde "natuurlijke taal" interfaces voor Intelligente Onderwijssystemen. To be published in Tennysson, R. (Ed.): Computers in Human Behavior, Summer issue.

Geest, Th.v.d. (1986). De computer in het schrijfonderwijs: perspectief voor een procesbenadering. Onderzoeksrapport Faculteit Toegepaste Onderwijskunde. Enschede, the Netherlands: Universiteit Twente.

Miller, J.R. (1988). The Role of Human-Computer Interaction in Intelligent Tutoring Systems. In M.C. Polson & J.J. Richardson: Foundations of Intelligent Tutoring Systems. Chapter 6, pp. 143-191. Hillsdale, New Jersey: Lawrence Erlbaum.

Oberem, G.E. (1991). The Development and Use of Aritificial Intelligence Techniques for the Delivery of Computer-Assisted Instruction on Advanced Computers. Research Report. Rhodes University, Grahamstown.

O'Brien, P. (1992). eL: using AI in CALL. Intelligent Tutoring Media, Vol. 3, No. 1, February 1992.

Pijls, F., Daelemans, W., & Kempen, G. (1987). Artificial Intelligence Tools for Grammar and Spelling Instruction. Instructional Science no. 16, pp. 319 - 336.

Pijls, F., & Sandberg, J. (1989). De Computer als Expert en Didacticus. Een introductie in het onderzoek naar intelligente onderwijssystemen. Muiderberg, the Netherlands: Coutinho.

Rypa, M. (1992). CALLE: A Computer-Assisted Language Learning Environment. In F.L. Engel, D.G. Bouwhuis, T. Bösser & G. d'Ydewalle (Eds.): Cognitive Modelling and Interactive Environments in Language Learning. Proceedings of the NATO Advanced Research Workshop in Advanced Educational Technology. Series F: Computer and Systems Sciences, Vol. 87., pp. 175 - 182. Berlin: Springer Verlag.

Schuster, E. (1985). Grammar as User Models. IJCAI '85, pp. 20 - 22.

Schuster, E., & Finin, T. (1986). VP^2: The Role of User Modelling in Correcting Errors in Second Language Learning. In A.G. Cohn & J.R. Thomas (Eds.), Artificial Intelligence and its Applications, pp. 197 - 209. Chichester: John Wiley & Sons.

Schwind, C.B. (1987). An Overview of an Intelligent Language Tutoring System. In Advances in Artificial Intelligence, Proceedings of ICAI 1986, pp. 189 - 205. London: Hermes.

Shapiro, S.C. (Ed.) (1987). Encyclopedia of Artificial Intelligence. New York: John Wiley & Sons.

Vosse, Th.G. (1989). De Bouw van een Schooltekstverwerker. Eindverslag SVO-project 56303.

Wenger, E. (1987). Artificial Intelligence and Tutoring Systems. Los Altos, CA: Morgan Kaufmann.

Winkels, R., Breuker, J. & Sandberg, J. (1988). Didactic Discourse in Intelligent Help Systems. In Proceedings ITS '88 Montreal, pp. 279 - 285.

Winkels, R. (1992). Explorations in Intelligent Tutoring and Help. PhD Thesis. Amsterdam: IOS Press.

Woolf, B. (1988). Intelligent Tutoring Systems: A Survey. In H. Schrobe and AAAI (Eds.) Exploring Aritificial Intelligence, Chapter 1, pp.1-43. Los Altos, CA: Morgan Kaufmann.

Woolf, B., & Cunningham, P.A. (1987). Multiple Knowledge Sources in Intelligent Teaching Systems. In IEEE Expert, Summer 1987, pp. 41-73.

Yazdani, M. (1988). Language Tutoring with Prolog. In Proceedings ITS'88 Montreal, pp.40-45.

Yazdani, M. (1989). An Artificial Intelligence Approach to Second Language Teaching. In Maurer (Ed.): Computer-Assisted Learning. Second International Conference Computer-Assisted Learning (ICCAL '89), pp. 618 - 624. Berlin: Springer Verlag.

The Probabilistic Retreat From Biases: Implications for Man Machine Communication?

Mike Oaksford[1]

[1]Department of Psychology, University of Wales at Bangor, Bangor, Gwynedd, LL57 2DG, UK.
(E-mail: PSSØ27@bangor.ac.uk OR mike@cogsci.ed.ac.uk)

1 Introduction

Until recently the psychological study of reasoning has been interpreted to reveal that untutored human reasoners fall far short of the standards of rationality provided by normative theories. In laboratory tasks normal adults appear to reveal a variety of apparently irrational and systematic biases from the prescriptions of logic and mathematics (see, Evans, 1989, or Evans, Newstead & Byrne, 1993, for overviews). Theoretical responses to these findings have included appeal to short cut processing strategies, i.e., heuristics, and the use of special representational formats, i.e., mental models. Over the last few years, however, it has become clear that many of the tasks taken to reveal biases are susceptible to more rational interpretations. It seems that the wrong logico-mathematical frameworks may have been taken to define rational behaviour in these tasks and that subjects may use probabilistic strategies that are wholly consistent with mathematical probability theory.

While I have little knowledge of man-machine communication or of the rationale behind work in this area, it seems that an awareness of these developments may be important to this endeavour. According to Norman and Draper (1986), in designing profitable human-computer interactions it is important that a task be "presented so that it matches human skills." The skills to be matched invariably involve some element of reasoning. Hence in constructing methodologies for interacting with computers a minimal requirement may be an understanding of how people reason. Moreover, on briefly scanning the contents of some recent edited collections in this area I was struck by how many made appeal to the theoretical constructs - such as heuristics and mental models - that have become popular as a result of work on biases in human reasoning. To the extent that the invocation of such constructs is being re-thought in the cognitive psychology of reasoning, a similar re-evaluation may be necessary in the area of man-machine communication.

In this chapter I discuss three areas where a probabilistic re-

interpretation has been successful: probabilistic reasoning, causal reasoning, and hypothesis testing. I first introduce each area, outlining the principle tasks used. I then discuss the new probabilistic accounts of these data due, in the case of probabilistic reasoning, to Birnbaum (1983) and Gigerenzer and Murray (1987), in the case of causal reasoning, to Cheng and Novick (1990), and in the case of hypothesis testing, to Oaksford and Chater (Oaksford & Chater, 1993).

2 Biases in Human Reasoning

2.1 Probabilistic Reasoning and Base Rate Neglect

Perhaps one of the most well known examples of bias in the reasoning literature concerns Kahneman and Tversky's (1972) demonstration that subjects tend to ignore base rate information. I illustrate this using Tversky and Kahneman's (1980) cabs problem. Subjects were provided with the following information:

> A cab was involved in a hit-and-run accident at night. Two cab companies, the Green and the Blue, operate in the city. You are give the following data:
>
> (i) 85% of the cabs in the city are Green and 15% are Blue
>
> (ii) A witness identified the cab as a Blue cab. The court tested his ability to identify cabs under appropriate visibility conditions. When presented with a sample of cabs (half of which were Blue and half of which were Green) the witness made correct identifications in 80% of the cases and erred in 20% of the cases.
>
> *Question*: What is the probability that the cab involved in the accident was Blue rather than Green.

In terms of Bayes' theorem the probability of the cab being blue given the witnesses testimony ($P(Blue|"Blue")$) is,

$$P(Blue|"Blue") = \frac{P("Blue"|Blue)P(Blue)}{P("Blue")}$$

That is the likelihood that the witness correctly identifies the cab ($P("Blue"|Blue)$) times the ratio of the base rates of Blue cabs in the city and the witness giving the evidence "Blue." The probability $P("Blue")$ can be calculated by conditioning on the probabilities of the cab being Green or Blue (since these are exhaustive and mutually exclusive events), so,

$$P("Blue") = P("Blue"|Blue)P(Blue) + P("Blue"|Green)P(Green)$$

The Cabs problem therefore provides all the information needed to deploy Bayes' theorem to calculate the probability that it was indeed a Blue cab that was involved in the hit-and-run accident:

$$P(Blue|"Blue") = \frac{0.80 \times 0.15}{0.80 \times 0.15 + 0.20 \times 0.85} = 0.41$$

However, most undergraduate student subjects report the probability of the cab being Blue as around 0.80 (this was the median response and the most frequent response). This concurs with the likelihood that the witness correctly identifies the cab, i.e., 0.80. Tversky and Kahneman therefore concluded that their subjects neglect base rate information in calculating posterior probabilities.

The explanation of these data provided by Tversky and Kahneman (1980) involves the causal relevance of the base rates provided. They argue that subjects do not view the base rates of cabs *in the city* as causally relevant to whether it was a Blue or Green cab that was involved in the accident. They tested this hypothesis by making the base rates more relevant by describing them as the base rates of cab *accidents* involving Blue and Green cabs. In this condition subjects report median probabilities of 0.6, i.e., closer to the predicted value. This explanation does not directly fit into the heuristic framework usually associated with Kahneman and Tversky. However, in the simulation heuristic (Kahneman & Tversky, 1982) causes can be assessed by undoing a cause in our minds and then seeing if the effect still occurs in a mental simulation. Thus their account in terms of causal relevance could be accommodated by the simulation heuristic (although nothing in my subsequent argument depends on this point).

2.2 Bias in Causal Attributions

Another area where bias has been commonly observed is causal attribution in social psychology. I illustrate these biases by reference to Kelley's (1967) ANOVA model which represents the attribution process as relying on the principle of co-variation embodied in the analysis of variance. The analysis of variance is of course typically regarded as a normative inductive procedure.

The co-variation principle states that for an event C to be viewed as the cause of event E, E must occur when C does and when C does not occur neither should E. Kelley proposed three dimensions in the attribution of the cause of an event. The cause could be due to persons (P), stimuli (S) or times (T). Three informational variables contribute to "who, what or when" gets the blame, these are:

Consensus between P's response to S and other's response to S at T.
Distinctiveness of P's response to S from P's response to other stimuli at T.
Consistency of P's response to S at T with P's responses to S at other Ts.

Only distinctiveness is directly proportional to covariation. So *high* distinctiveness corresponds to a *high* covariation between S and the response, i.e.,

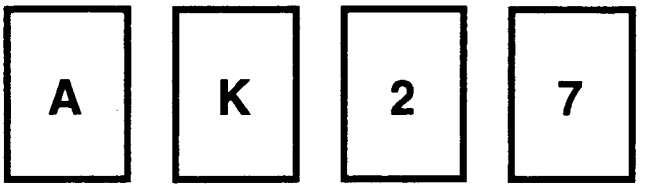


Figure 1: The four cards in the abstract version of Wason's selection task.

high distinctiveness means the response rarely occurs in the absence of S. This contrasts with, for example, consensus information, where *high* consensus corresponds to a *low* covariation between P and the response, i.e., lots of other people also make this response. It is then a trivial corollary of the model that the following patterns of high or low consensus, distinctiveness or consistency information (in that order) should lead to the indicated attributions: **LLH** = Person, **HHH** = Stimulus, **HLL** = Time. However, while there has been some good general agreement between model and data, some systematic biases have also been observed.

I concentrate on two of these biases. First, it has been found that consensus information is underused particularly in determining person (P) attributions. Recall that low consensus means that there is a high covariation between that person and the response, i.e., the response is rarely observed without this person being involved. Consensus information should therefore be important in determining person attributions. However, McArthur (1972) found that consensus information accounts for only 6% of the variance in person attributions and only 3% of the total variance in causal attributions. Second, there would appear to be a strong bias towards person attributions as opposed to stimulus or time attributions. In McArthur's (1972) results (noted by Jaspars *et al.*,1983), 82% of subjects made person attributions when P covaried with the response, as opposed to 62% who made stimulus attributions when S covaried with the response, and 33% who made time attributions when T covaried with the response.

2.3 Hypothesis Testing

Perhaps the most notorious area where biases have emerged in the reasoning literature concerns human hypothesis testing and in particular work carried out using Wason's (1966, 1968) selection task. Wason's task, which is probably the most replicated task in cognitive psychology, requires subjects to assess whether some evidence is relevant to the truth or falsity of a conditional rule of the form *if p then q*, where by convention "*p*" stands for the antecedent clause of the conditional and "*q*" for the consequent clause. Subjects are presented with four cards each having a number on one side and a letter on the other (see Figure 1).

The subjects are also given a rule, e.g., *if there is a vowel on one side* (*p*), *then there is an even number on the other side* (*q*). The four cards show an "A"(*p* card), a "K"(*not-p* card), a "2"(*q* card) and a "7"(*not-q* card). By convention these are labelled the *p* card, the *not-p* card, the *q* card, and the *not-q* card respectively. Subjects have to select those cards they must turn over to determine whether the rule is true or false. Typical results were: *p* and *q* cards (46%); *p* card only (33%), *p*, *q* and *not-q* cards (7%), *p* and *not-q* cards (4%) (Johnson-Laird & Wason, 1970).

The normative theory of the selection task was derived from standard logic and Popper's (1959) account of falsification. Popper argued that a scientific law can not be shown to be true because it is always possible that the next instance of the law observed will be falsifying. However, one can be logically certain that a law is false by uncovering a single counter-example. Furthermore, this means that scientific reasoning is fundamentally deductive in character - what must be established is a logical contradiction between putative laws and observation. Hence looking for false (*p* and *not-q*) instances should be the goal of scientific inquiry. However, in the selection task subjects typically select cards that could *confirm* the rule, i.e., the *p* and *q* cards.

Confirmation bias, however, does not adequately account for the data on the selection task. Evans and Lynch (1973) showed that when negations are varied between antecedent and consequent of a rule subjects reveal a bias towards selecting those cards that are named in the rules, ignoring the negations. For example, take the rule "if there is not an A on one side, then there is not a 2 on the other side", and the four cards showing an "A", a "K", a "2" and a "7". The matching response would be to select the A and the 2 cards. In contrast according to confirmation the K and the 7 card should be selected and for falsification the K and the 2 card should be selected. Evans and Lynch (1973) refer to this phenomenon as *matching bias*. Matching could also account for the data from the standard affirmative selection task (see above) where no negations are employed in the task rules. This is because the matching and confirmation strategies coincide on the selections they predict for the affirmative rule, i.e., the A and the 2 cards. Thus subjects behaviour on the selection task may not reveal any hypothesis testing strategy at all!

3. Interlude

Having summarised three areas where biases have typically been observed in human reasoning I now turn to how these "biases" are being reconciled with normative theory by the adoption of more appropriate normative models based largely on probability theory. It is interesting to first note that there were always three possible reactions to the observation of biases (Thagard, 1988, p. 123):

1. *People are dumb*. They simply fail to follow the normatively appropriate

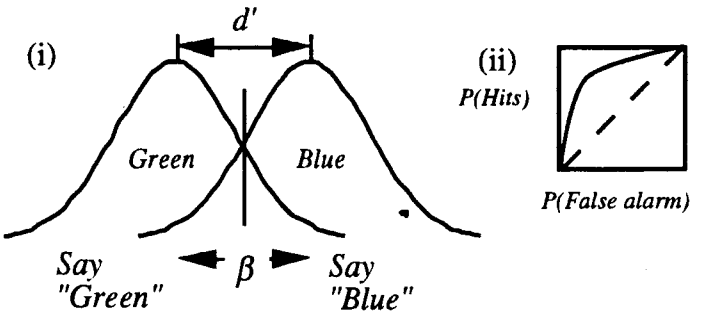


Figure 2: Signal detection theory. (i) Area in "Blue" distribution to the right of β = *P(hit)*; area in the "Green" distribution to the right of β = *P(false alarm)*. (ii) The Receiver Operating Curve (ROC)

inferential rules.
2. *Psychologists are dumb.* They have failed to take into account all the variables affecting human inferences, and once all the factors are taken in to account it should be possible to show that people are in fact following appropriate rules.
3. *Logicians [Mathematicians,* my insertion*] are dumb.* They are assessing the inferential behaviour of human thinkers against the wrong set of normative standards.

Psychologists seem to have largely opted for 1 and explained why in terms of cognitive limitations. My preference is go after 2 and 3 However, my target in this paper is mainly 3 and I will only briefly touch on work pursuing 2 when I discuss Cheng and Novick's account of biases in causal attribution. (For work pursuing 2, see, e.g., Beyth-Marom, 1982; Gigerenzer, Hell & Blank, 1988; Oaksford & Stenning, 1992.)[1]

4. The Probabilistic Retreat from Biases

4.1 Probabilistic Reasoning and Base Rate Neglect.

Birnbaum (1983) has shown that the cabs problem of Tversky and Kahneman (1980) can be recast in the language of Signal Detection Theory (SDT) which is based on Neyman and Pearson's statistics. He shows that when this is done the resulting model concurs with Tversky and Kahneman's (1980) data.

SDT represents the discrimination problem in terms of two partially overlapping normal distributions along which a decision boundary or criterion (β) may be specified (see Figure 2(i)). The discriminability of the two distributions is determined by the separation between their mean (peak) values,

[1] It is worth noting that Logicians/Mathematicians do not always see their work as directly applicable to these data in the ways that psychologists would wish. Hence often it is simply the wrong theory that psychologists have "adopted."

this is d' which is measured in z-score units. In the case of the cabs problem one of the distributions corresponds to Green and the other to Blue. The eye witnesses testimony defines a hit rate (80%) and a false alarm rate (20%) from which d' can be calculated. In this case it equals 1.68. Given one point, a whole ROC curve is defined which corresponds to a plot of pairs of false alarm rates and hit rates for a single d' as the criterion β is varied (see Figure 2(ii)). The standard psychophysical procedure for plotting an ROC curve is to vary the base rates of the distributions (Green and Blue) thereby moving the criterion to obtain another false alarm rate/hit rate pair. The critical observation in accounting for the cabs problem is that the base rates in the court's test (50%-50%) were different from the night when the witness saw the accident (85%-15%). Hence the witnesses criterion will have been set differently on the night of the accident in comparison to when the test was conducted. However, the lighting conditions were carefully mimicked in the test so it can be assumed that d' remains the same.

What is the effect of altering the base rates? The problem is that SDT does not specify where the criterion is to be placed when the base rates change. However, Birnbaum assumes that witnesses set their criterion to minimise incorrect testimony in the knowledge that there are 15% Blue cabs. This leads to a hit rate of 0.302 and a false alarm rate of 0.698. If these values are then translated back in to likelihoods and used in Bayes' theorem, a value of 0.82 is returned for the posterior probability that the cab was indeed blue given the witnesses testimony. This is remarkably close to the 0.80 actually observed in Tversky and Kahneman (1980).

As Gigerenzer and Murray (1987) point out this analysis depends on *attending* to bases rates not *neglecting* them. They also observe that just by taking Tversky and Kahneman's own proposals seriously with respect to which base rates should be used can lead to the observed data. Subjects may regard the base rates *in the city* as irrelevant seeking instead base rates of cabs *in accidents*. In the absence of relevant information to the contrary, they may assume that the Green and Blue cabs are equally likely to be involved in an accident, i.e., the relevant base rates are 50%-50%. Assuming these base rates, the posterior probability that the cab was indeed blue given the witnesses testimony is 0.80! In sum, it seems that the conclusion of a bias towards base rate neglect may have been premature.

This account of the cabs problem has probably not been influential because (i) of its isolation in a sea of data apparently supporting the existence of biases and (ii) the corresponding lack of promising normative analyses of these data. However, such analyses are now increasing in number as our next two more recent examples demonstrate.

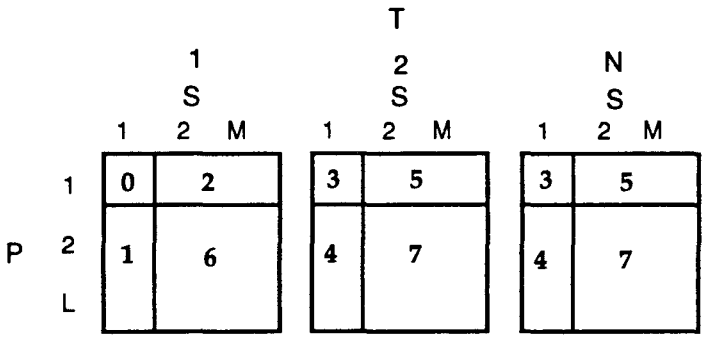


Figure 3. Kelley's ANOVA cube for persons (P), stimuli (S) and times (T), showing the eight regions of information.

4.2 Bias in Causal Attributions

Cheng and Novick (1990) argue that the problem of causal attribution can be framed in terms of their normative probabilistic contrast model. They show that their model predicts the detailed pattern of results on causal attribution when subjects are provided with complete information.

Cheng and Novick (1990) note that only incomplete information is provided by consensus, distinctiveness and consistency. If we cast the persons, stimuli and times dimensions into Kelley's famous ANOVA cube then each of these informational sources can be treated as specifying information about specific regions of the cube (see Figure 3).

Figure 3 shows the eight regions of the cube. Consensus information provides information about the agreement between person 1's response to stimulus 1 at time 1, i.e., it provides information about region 1 of the cube (distinctiveness provides information about region 2 and consistency about region 3). Consensus does not provide information about the agreement between person 1's response and others' responses to the other stimuli S_M - S_1 at other times T_N - T_1, i.e., it fails to provide information about regions 4 and 5 of the cube. Cheng and Novick (1990) note that the ANOVA uses information in all regions to determine main effects and interactions. Hence biases in causal attribution may be due to incomplete information. They therefore used materials that provide the missing information. For example, the following materials were used for an **LLH** problem (see above).

Jane had fun washing the dishes on this occasion.
1. In fact, Jane always has fun doing all chores.
2. But nobody else ever has fun doing any chores.

The first sentence provides information about region 0, i.e., there is a positive relationship between this person (Jane) on this occasion for this stimulus

(washing the dishes) and having fun (the response). Clause 2 provides information about regions 2, 3 and 5 (+ relationship) and clause 2 provides information about regions 1, 4, 6 and 7 (- relationship).

Cheng and Novick (1990) derive predictions for the detailed patterns of attributions (including interactions) using their probabilistic contrast model which provides a simpler way of computing co-variation than the complex quantitative calculations required by the ANOVA. A contrast Δp_i for a causal factor i is defined as:

$$\Delta p_i = p_i - p_{\bar{i}} \qquad (1)$$

Where p_i is the proportion of times that the effect occurs when i occurs and $p_{\bar{i}}$ is the proportion of times that the effect occurs when i does not. Sufficiently large non-zero values of Δp_i (to some pre-set criterion) indicate the i is a cause (positive or negative) of the effect. Interaction effects can also be defined as follows:

$$\Delta p_{ij} = (p_{ij} - p_{\bar{i}j}) - (p_{i\bar{j}} - p_{\bar{i}\bar{j}}) \qquad (2)$$

The interpretation of (2) is straightforward. p_{ij} is the proportion of times the effect occurs when both causal factors i and j are present; $p_{\bar{i}j}$ is the proportion of times the effect occurs when i is absent and j is present; and so on. A sufficiently large non-zero difference between these contrasts reveals a (positive or negative) two way interaction. In contrast to the ANOVA, the sign of a contrast in the probabilistic contrast model provides a direct indication of the direction of an effect.

Cheng and Novick (1990) show that their model predicts the detailed pattern of results on causal attribution when complete information is specified (like the ANOVA complete information is required for their model). Moreover, their data reveal that with complete information all signs of bias evaporate. Taking the bias against using consensus information, Cheng and Novick (1990) found that consensus accounted for 31% of the variance in their data, distinctiveness 33%, and consistency 24%, i.e., no indication of bias was observed. Similarly, for the bias towards person attributions they found that for the problems where only persons (**LLH**), stimuli (**HHH**), or times (**HLL**) covaried with the response the proportion of subjects making the appropriate attribution were 75%, 80% and 85%, i.e., no indication of bias was observed.

Again conclusions of bias have proved to be premature, when complete information is provided subjects show no bias against using consensus information or towards making person attributions. Moreover, their behaviour is predicted by a normative probabilistic model.[2]

---

[2] It could be argued that Cheng and Novick's probabilistic contrast model is redundant if it makes the same predictions as the ANOVA. However, (i) their model also accounts for

4.3 Hypothesis Testing

Oaksford and Chater (1993) show that Wason's selection task can be recast as a problem of optimal data selection that is susceptible to a Bayesian analysis. When this is done they show that all the main results on the selection task can be derived from their normative model.

Any problem of deciding what experiment to perform next, or which observation is worth making, is a problem of optimal data selection. Consider assessing the truth of a mundane regularity, such as that eating tripe makes people feel sick: to collect evidence should we ask people who complain that they feel sick, or perhaps those who do not feel sick, whether they have eaten tripe; should we investigate people who are known tripe eaters or tripe avoiders. This case is analogous to the selection task: the rule *if tripe then feeling sick* is of the form *if p then q*, and the various populations which may be investigated correspond to the various visible card options, *p*, *not-p*, *q* and *not-q*. The subject must judge which of these people should be questioned concerning either how they feel or what they have eaten.

Oaksford and Chater (1993) suggest that those data points expected to provide the greatest information gain should be selected. The information gain given some data is the difference between the uncertainty *before* receiving the data and the uncertainty *after* receiving the data. Uncertainty is measured using the Shannon-Wiener information measure and hence the information gain of a data point D may be defined as follows:

$$I_g(D) = -\sum_{i=1}^{n} P(H_i)\log_2 P(H_i) + \sum_{i=1}^{n} P(H_i|D)\log_2 P(H_i|D) \qquad (3)$$

That is the difference between the information contained in the *prior* probability of a hypothesis (H_i) and the information contained in the *posterior* probability of that hypothesis given some data D. In calculating the posterior probability Oaksford and Chater assume that the hypothesis under test, which indicates a dependence between p and q, is compared to an alternative hypothesis which indicates that p and q are independent (the principle of indifference between these hypotheses is assumed, so they are both taken to be equally probable, i.e., the *priors* for both are 0.5).

In the selection task, however, when choosing which data point to examine further (that is, which card to turn), the subject does not know what that

---

other areas of causal reasoning where bias had been observed, (ii) the computations involved in their model are far simpler and therefore more psychologically plausible, (iii) a direct indication of facilitatory or inhibitory causation is provided.

data point will be (that is, what will be the value of the hidden face). However, what can be calculated is the *expected* information gain. If we assume that there are two possible data points that might be observed (in the case of the *not-q* card, for example, corresponding to the hidden face being, say, p or *not-p*), then the expected information gain of the *not-q* card is:

$$E(I_g(\neg q)) = \sum_j P(H_j)P(p_t|\neg q, H_j)I_g(p_t, \neg q) \qquad (4)$$

Oaksford and Chater then show that arriving at appropriate probability models for the dependence and independence hypotheses only involves specifying the frequency in the population of the properties described by p and q. They label these parameters a and b respectively. With a and b specified all the values needed to compute (4) for each card are available.

In accounting for the selection task Oaksford and Chater (1993) carried out an extensive meta-analysis of the data which revealed that over some 34 studies involving almost 900 subjects the frequency of card selections was ordered such that $p > q >$ *not-q* > *not-p*. To model this data they calculated the expected information gain of each card with $a = b = 0.1$. That is, they assumed that the properties described in p and q are rare in the population. A similar assumption was made by Klayman and Ha (1987) in accounting for related data on Wason's (1960) 2-4-6 task. On this assumption the order in expected information gain is:

$$E(I_g(p)) > E(I_g(q)) > E(I_g(not-q)) > E(I_g(not-p))$$

That is, if subjects base their card selections on the expected information gain of the cards then the observed order in the frequency of card selections is exactly as Oaksford and Chater's (1993) model of optimal data selection predicts. Oaksford and Chater (1993) also show how their model generalises to all the main patterns of results in the selection task, including the manipulation described above of introducing negations into the task rules. In sum, if the selection task is regarded as a problem of probabilistic optimal data selection then the putative biases observed in this task disappear.

5 Conclusions and Some Possible Implications for Man-Machine Communication

5.1 Summary

I began this chapter by claiming that much of the bias literature needs to be re-interpreted in the light of the growing number of normative probabilistic re-analyses of the data upon which claims of bias have been based. These analyses,

as the first example of Birnbaum (1983) reveals, were often contemporaneous with the heyday of the "heuristic and biases" tradition. However, these analyses were largely set aside in the rush to explain these biases in process terms. Of course these accounts simply ignored the fact that they were now assigning apparently irrational functions to the processes they invoked. Recently, however, there has been a re-emergence of concern with rationality and rational analysis (see, e.g., Anderson, 1991; and the collection edited by Manktelow & Over, 1993) and with it I suspect that we are witnessing a degenerative problem shift (Lakatos, 1970) for work in the area of heuristics and biases. Lopes (1991) has observed that the rhetorical emphasis of the heuristics and biases tradition was on Thagard's 1 above, i.e., on human dumbness. She points out that this de-emphasised the importance of rationality in cognitive explanation. In witnessing a problem shift back towards rational function we should be wary that the psychological insights of the heuristics and biases account are not similarly abandoned. As Gigerenzer and Murray (1987, see also, Gigerenzer, Hoffrage & Kleinbölting, 1991) observe there has always been an essential subjective, psychological aspect to probability theory. As we have seen assumptions about which *priors* to employ or their effects on the decision criterion, for example, are not decideable within probability theory. In arriving at psychological theories of reasoning in any domain there will be a need for essentially psychological accounts of how these values are arrived at and how people tractably implement their probabilistic competence. Nevertheless, following Marr (1982), first identifying the rational functions that the human cognitive system needs to compute is again being seen as a feasible and necessary first step in specifying an adequate cognitive theory.

5.2 Man-Machine Communication

While I am not competent to judge whether these developments will have immediate ramifications within man-machine communication, I can outline an area where these considerations may be relevant: the apparent success of graphics interfaces like the highly usable Macintosh interface. I make the assumption that all interfaces are deterministic systems. That is, the actual rules that determine what happens given user input are deterministic, i.e., as long as the system is functioning normally, a given input, key press sequence, mouse click in a menu location etc., yields a single specific action with probability 1. Now at some metaphysical level of description the world may be a deterministic system. Certainly classical mechanics which for the bulk of our everyday macroscopic needs is all we need to predict the physical world, regards the world deterministically. However, this does not mean that in acquiring knowledge of the macroscopic world human beings model it deterministically. Irremediable ignorance of the disposition of a complex system whose next state we wish to predict often precludes deterministic prediction. Instead scientists must settle for

probabilistic theories in order to make adequate predictions. Similarly, when confronted with a complex user interface novice users may model the system not deterministically but probabilistically.

This provides a possible explanation for the success of interfaces like the Macintosh. Deterministic command line interfaces are all or nothing affairs, either you know the appropriate command or you don't. The organisation of access is also very flat in that if you don't know the specific command there are no higher level exploratory actions you can perform in order to find out. This kind of organisation is only suited to rote learning of the appropriate command language and rules. This contrasts with the Macintosh interface, where there are often many ways of performing the same action. Moreover, there are a set of high level actions, like pulling down menus, clicking on *labelled* buttons, scrolling etc. that enable the user to interact with the computer in an exploratory manner. This is analogous to our normal inductive interactions with the world. In acquiring practical skills the ability to manipulate and explore bits of our world permits us to learn much faster. It seems possible, therefore, that these interfaces succeed because they exploit the same abilities that allow us to learn so effectively about the world in order to inductively explore the interface. In sum, novices may model their interactions probabilistically in accordance with their natural competences; the very competences that are now being revealed in the probabilisitic retreat from bias.

References

Anderson, J. R. (1991). Is human cognition adaptive? *Behavioral and Brain Sciences*, **14**, 471-517.

Beyth-Marom, R. (1982). Perception of correlation reexamined. *Memory & Cognition*, **10**, 511-519.

Birnbaum, M. H. (1983). Base rates in Bayesian inference: Signal detection analysis of the cab problem. *American Journal of Psychology*, **96**, 85-94.

Cheng, P. W., & Novick, L. R. (1990). A probabilistic contrast model of causal induction. *Journal of Personality and Social Psychology*, **58**, 545-567.

Evans, J. St. B. T. (1989). *Bias in human reasoning: Causes and consequences*. Hillsdale, NJ: Erlbaum.

Evans, J. St. B. T., & Lynch, J. S. (1973). Matching bias in the selection task. *British Journal of Psychology*, **64**, 391-397.

Evans, J. St. B. T., Newstead, S. E., & Byrne, R. M. J. (1993). *Human Reasoning*. Hillsdale, N.J.: Lawrence Erlbaum Associates.

Gigerenzer, G., Hell, W., & Blank, H. (1988). Presentation and content: The use of base rates as a continuous variable. *Journal of Experimental Psychology: Human Perception and Performance*, **14**, 513-525.

Gigerenzer, G., Hoffrage, U., & Kleinbölting, H. (1991). Probabilisitic mental models: A Brunswickian theory of confidence. *Psychological Review*, **98**, 506-528.

Gigerenzer, G., & Murray, D. J. (1987). *Cognition as intuitive statistics*. Hillsdale, N.J.: Lawrence Eerlbaum Associates.

Jaspars, J. M. F., Hewstone, M. R. C., & Fincham, F. D. (1983). Attribution theory and research: The state of the art. In J. M. F. Jaspars, F. D. Fincham, & M. R. C.

Hewstone (Eds.), *Attribution theory: Essays and experiments*, (pp. 3-36). Orlando, FL: Academic Press.
Johnson-Laird, P. N., & Wason, P. C. (1970). A theoretical analysis of insight into a reasoning task. *Cognitive Psychology*, **1**, 134-148.
Kahneman, D., & Tversky, A. (1972). On prediction and judgement. *ORI Research Monograph*, **12**.
Kahneman, D., & Tversky, A. (1982). The simulation heuristic. In D. Kahneman, P. Slovic, & A. Tversky (Eds.), *Judgement under uncertainty: Heuristics and biases*, Cambridge: Cambridge University Press.
Kelley, H. H. (1967). Attribution theory in social psychology. In Levine D. (Ed.), *Nebraska Symposium on Motivation* Vol. 15, (pp. 192-238). Lincoln: University of Nebraska Press.
Klayman, J., & Ha, Y. (1987). Confirmation, disconfirmation and information in hypothesis testing. *Psychological Review*, **94**, 211-228.
Lakatos, I. (1970). Falsification and the methodology of scientific research programmes. In I. Lakatos, & A. Musgrave (eds.), *Criticism and the growth of knowledge*, (pp. 91-196). Cambridge: Cambridge University Press.
Manktelow, K. I., & Over D. E. (Eds.) (1993). *Rationality*, London: Routledge.
Marr, D. (1982). *Vision*. San Fransisco, CA: W. H. Freeman & Co.
McArthur, L. Z. (1972). The how and what of why: Some determinants and consequecnces of caual attribution. *Journal of Personality and Social Psychology*, **22**, 171-193.
Norman, D. A., & Draper, S. W. (1986). *User-centred System Design*. Hillsdale, N.J.: Lawrence Erlbaum Associates.
Oaksford M, & Chater N. (1993). A rational analysis of the selection task as optimal data selection. Technical Report No. CCN-UWB-93-05: Centre for Cognitive Neuroscience, University of Wales, Bangor. (In submission).
Oaksford, M., & Stenning, K. (1992). Reasoning with conditionals containing negated constituents. *Journal of Experimental Psychology: Learning, Memory & Cognition*, **18**, 835-854.
Popper, K. R. (1959). *The logic of scientific discovery*. London: Hutchinson.
Thagard, P. (1988). *Computational philosophy of science*. Cambridge, MA: MIT Press.
Tversky, A., & Kahneman, D. (1980). Causal schemas in judgements under uncertainty. In Fishbein M (Ed.), *Progress in social psychology*, Hillsdale, N.J.: Lawrence Erlbaum Associates.
Wason, P. C. (1960). On the failure to elminate hypotheses in a conceptual task. *Quarterly Journal of Experimental Psychology*, **12**, 129-140.
Wason, P. C. (1966). Reasoning. In B. Foss (ed.), *New horizons in psychology*, Harmondsworth, Middlesex: Penguin.
Wason, P. C. (1968). Reasoning about a rule. *Quarterly Journal of Experimental Psychology*, **20**, 273-281.

An X Window Based GUI for the Dynamic Simulation of the Chemical Recovery Cycle of a Paper Pulp Mill

F. Pais[1], B. Gay[2] and A. Portugal[1]

[1]Department of Chemical Engineering, University of Coimbra, Largo Marquês de Pombal, 3000-Coimbra, Portugal
[2]Department of Computer Science and Applied Mathematics, Aston University, Birmingham B4 7ET, U.K.

Abstract. This contribution describes an interactive software system (XProcSim) for easy and efficient analysis of the behaviour in transient state of the chemical recovery cycle of a pulp mill. It is formed by two independent but coordinated components: a graphical user interface (GUI), written in C, which was developed using the X Window system together with Sun's XView toolkit, and simulation modules written in FORTRAN77, which may run on the same or on a different machine. Process representation, control of the integration performed by the simulation programs, data and perturbation inputs, and graphical representation of results are performed on a colour UNIX Sun SPARCstation2 + graphics accelerator using the mouse/menu driven interface. The keyboard is used to input data only (numeric values or character strings). Significant differences between previous systems and the software described here are XProcSim's use of the X Window system for the GUI development, which ensures portability to a number of hardware platforms. Remote, networked X-based display systems may be used, and there is also the possibility of running the simulation program on a remote machine. As the complexity of mathematical models increases, this is a critical difference for an on-line system as the execution time - and therefore the response time - may become prohibitive without the use of powerful machines, which do not necessarily possess graphics display capabilities. The distributed structure makes it possible to take the best possible advantage of existing machines.

Keywords. Graphical User Interface (GUI), Distributed Systems, Process Simulation, Chemical Recovery Cycle, Pulp Mill

1 Introduction

Process simulation is one of the most widely used tools for the design, optimization or simply the study, in steady or transient states, of industrial units and plants. If the analysis of process dynamics under conditions of start-up, grade changes, emergency outages and shutdowns is desired, then dynamic simulators are necessary (Schewk and others,1991). Our primary objective was to develop a package for the computer simulation of the transient state

of a paper pulp mill. Rather than describing mathematically the full set of industrial equipment involved such as pumps, valves, etc., we wanted to develop reliable dynamic models for the main units in the cycle and study their interaction. This is not a minor task as we are interested in the detailed behaviour of every unit which led, in some cases, to complex mathematical models of large dimension which present solution difficulties and whose simulation is time consuming (e.g., the lime kiln model alone is composed of a set of hyperbolic partial differential equations which, after spatial discretization and using the method of lines, is converted to hundreds of stiff ordinary differential equations). On the other hand, the migration of simulation tools from "simulation experts" into the hands of practical process engineers will be accelerated by the availability of user-friendly interfaces (Schewk and others,1991). In order to manage the amount of data involved and results produced, as well as to easily define or alter all the simulation parameters and impose the desired perturbations, it soon became apparent that a powerful user interface was needed. A number of GUI's have recently been developed for systems that range from genetics applications (Drury and others,1992) to economic analysis (Jensen and King,1992) and processing spectral data (King and Horlick,1992). The development of a GUI, which would enable us to display graphically the results of the simulation, was therefore a very important feature for the usability of our simulation programs.

2 On graphical user interfaces

"Software that functions but is difficult or confusing to use rarely actually gets to do anything"(Rimmer,1992). Based on software ergonomic investigations, the conclusion that graphical user interfaces are the interactive man/machine communication of the future is now widely accepted (Encarnação and others,1991). Recent studies have shown that users in a GUI environment work faster, more accurately and with lower frustration and fatigue levels than users in a text-based environment. Considerable increases in both productivity and accuracy of completed work are achieved using GUI's (Peddie,1992). There is, however, a lack of standardization and the number of existing systems is high. Examples are X Window for UNIX workstations, MS Windows and Presentation Manager for IBM Personal Computers, the MacIntosh software for Apple computers, GEM for the Atari and Intuition for the Amiga (Nicholls,1990; Pangalos,1992). In the UNIX-based world the *de facto* standard, developed by MIT (Scheifler and Gettys,1986), is usually known as X Window. It is not a GUI, but rather a common windowing system across networks connecting machines from different vendors, as well as a foundation on which to build graphical user interfaces (Peddie,1992). X can display output windows from several applications at once in one screen, and these programs can run simultaneously on different machines. The machines do not have to be the same type or run the same operating system (Moore,1990). Since the

X Window system is in the public domain and not particular to any hardware platform or operating system, and its distribution is not subject to the commercial interests of a single supplier, it may well become the standard windowing environment of the 1990's for all sorts of computers. Any machine that supports multitasking and interprocess communication and has a C compiler can support X (Moore,1990). One of the goals of X was to impose no style of interface on its users. It relies, in practical terms, on the development of graphical user interface toolkits - sets of widgets, or building blocks, such as menus, buttons, scrollbars, etc. - each with a consistent appearance and behaviour, to assist the application developer do his work. Initially developed as user interfaces were the Open Look GUI, by AT&T and Sun, and the Motif GUI by Open Software Foundation (OSF) and a number of soft and hardware manufacturers (DEC, Hewlett-Packard, Microsoft, etc.). These user interfaces were later complemented by toolkits, e.g. XView developed by Sun, and tend to become the standards for the UNIX world. Both of them possess a high degree of applications portability. In our work, we used a colour UNIX Sun SPARCstation2 + graphics accelerator workstation. Although it need not be so, in this case it functioned as both the client and the server of the X Window architecture (for details, see Reiss and Radin, 1992). We also used Sun's XView toolkit, together with the Open Look interface and window manager as developed by Sun and AT&T. A toolkit is an indispensable tool for saving time, since it avoids lengthy sets of low level intructions to create a simple object and guarantees consistency across all objects - that is, the same look and feel. Although functional consistency still has to be ensured by the interface designer, appearance consistency is ensured by the correct use of such a toolkit. The word *object* has been used here for the first time. To simplify developing applications, X toolkits employ concepts used in object-oriented programming. This is quite a feat as the toolkit is written in and for the C language, which does not directly support objects (Miller,1990).

3 Basic Features of XProcSim

XProcSim's interface was designed from scratch so a number of ways of achieving the desired objectives was usually available. Although "design is a creative process and hence there are few absolutes that must be adhered to" (Rubin,1988) consideration was given at every stage to the human factors guidelines which are, at present, playing an important role in the new science of user interface design. Two important characteristics of the interface we are developing are the fact it is menu-based and, primarily, a colour application. The advantages of menu-based user interfaces over command language ones are well known (Kantorowitz and Sudarsky,1989): no need to learn the command language which allows for productivity in a very short time, possibility of exploring the operations provided simply by browsing through the

menus and, if an adequate help system is installed, no need for a manual in most cases. The mouse is used for all the selection operations, whereas the keyboard is used to enter numeric values or character strings only.

On the other hand, colour is an excellent means of improving the comprehensibility of information displays such as pie and bar charts (Rubin,1988). However, use of colour must be judicious to avoid visual interaction that functions as communication-damaging noise and can produce strong after-effects. According to cartographic principles, background dull colours (muted, grayish or neutral) are more effective, allowing the smaller, bright areas to stand out with greater vividness (Tufte,1992). As to the palette of colours that should appear on the screen, a way to avoid "colour junk" is to use colours found in nature, chosen from widely spaced points in the colour spectrum (Tufte,1992; Rubin,1988). Rubin also states that the use of colour in a man-machine interface should be as consistent as possible with everyday usage. These basic principles were followed by XProcSim, where the default background colour is consistent through all the windows (light grey). Wherever possible, colour is used to convey information about the physical system itself. Units where green liquor is processed are coded green, units where lime is the main reactant are coded yellow, and so forth. In the lime kiln diagram, for example, colour goes from yellow to red as temperature increases. However, a commercial package should never be totally dependent on the use of colour since potential users would be significantly restricted and compatibility of the interface with monochrome displays is now under development. Another basic feature of XProcSim is the fact that all its windows are scalable which allows the user to resize them at will. No limitation has been imposed on the maximum number of windows which can be displayed simultaneously. Windows can at any time be reduced to their iconic state - icons have been attributed to all of them - so that avoidance of a cluttered screen is left to the user.

The structure of the interface is presented in Fig.1.

Basically, the objectives we aimed at were:

⋆ Graphical representation of the process and individual units ⋆ Organization and easy access to simulation data ⋆ Graphical representation of results ⋆ Possibility of executing the simulation program on remote machines and full control of its execution status ⋆ Easy access to integration data ⋆ On-line help system ⋆ Error warning and recovery ⋆ Possibility of executing a selected set of UNIX commands through menu options provided by the interface.

3.1 Graphical representation of the process

A very simplified flowsheet of the process, in the form of a block diagram, is displayed on request and gives acess to all the data for all the units. This diagram is formed by unit and stream type elements. Selecting one of these will display a pop-up menu containing all the information items available for that unit or stream. These include schematical representations of all the units

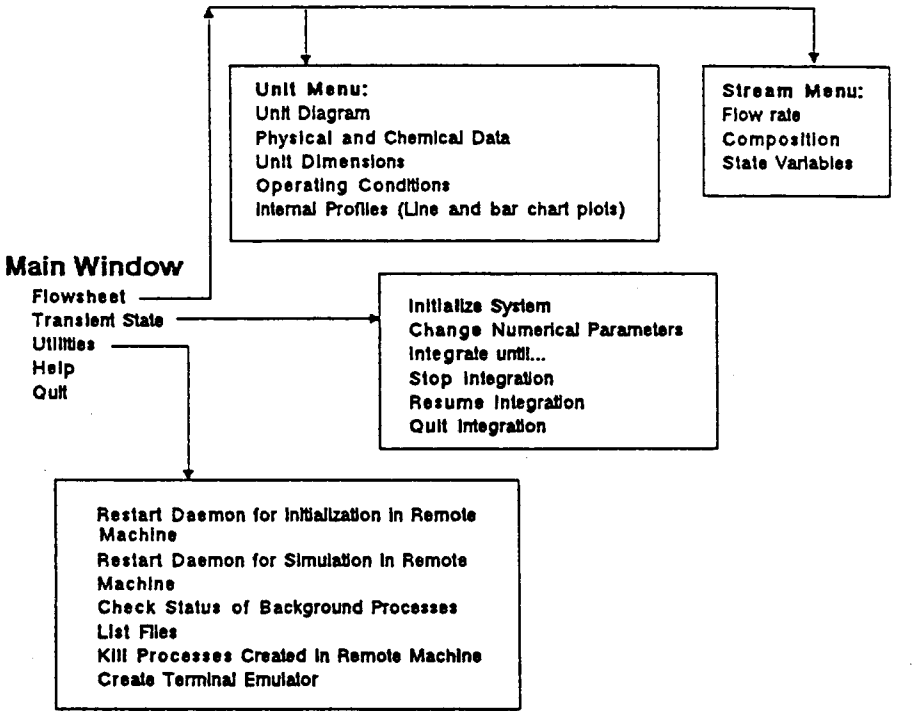


Fig. 1: Main options available through the interface.

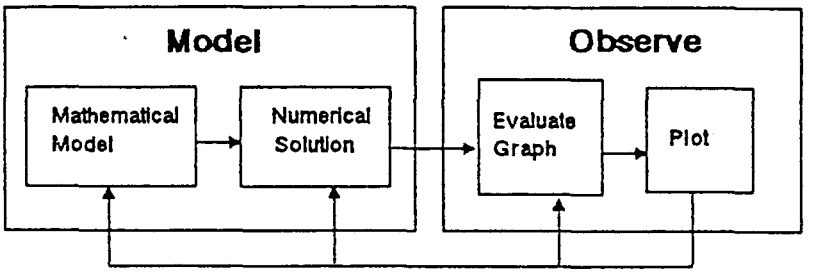


Fig. 2: Information Flow Model (Adapted from Nielsen,1991).

(which we hope will be of interest to the novice user), possibility of displaying instant internal profiles, design characteristics of the units, physico-chemical properties, etc..

3.2 Organization and easy access to simulation data

No commercial physical properties database was used. However, all the values used in the simulation of a unit, initially stored in files, are available and can be changed through the interface using data panels. Among these data we include the already referenced dimensions of the unit and physico-chemical parameters such as heat transfer coefficients, activation energies and Arrhenius constants for the chemical reactions, etc..

3.3 Graphical representation of results

Graphical representation of simulation results corresponds to four stages (see Fig.2).

Two interactive modules, one for line charts and one for bar charts, have been developed to perform the final task of creating an actual image on the screen. The line chart is used to represent either profiles along space for a given time or variation of a variable along time. Because it would be impossible to prepare charts with all possible combinations of variables, only a limited number have been selected and are automatically prepared for graphical display. However, both the line and bar chart modules can load data files so the more experienced user can generate his own data files corresponding to whatever variables he wishes to represent graphically. Both modules present a range of options to the user such as colour of all the elements, text font size, line width, colour and style (for the line chart), bar colour and fill pattern (for the bar chart), automatic or manual selection of axes range, etc..

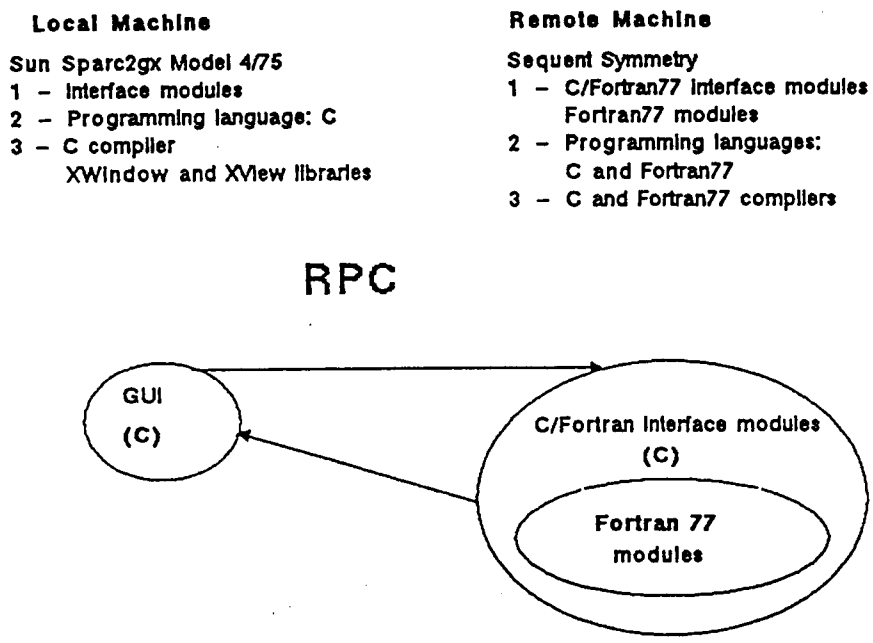


Fig. 3: C/FORTRAN77 structure.

3.4 Possibility of running the simulation program in a remote machine

As the complexity of mathematical models increases, so does usually execution time. If on-line interaction is desired, then for practical reasons very powerful machines may become necessary. The benefits of parallel processing for complex systems have become obvious in the last few years (Shandle,1990). The simulation program is at present being executed in a Sequent Symmetry UNIX machine, so execution times can reach critical values. Communication of the interface with the remote simulation program is done in two ways:

3.4.1 Successful RPC calls

Whenever possible, direct transfer of parameters is made. The RPC (Remote Procedure Call) package is used to link the local C interface routines to a remote C routine which in turn calls the remote FORTRAN77 routine (see Fig.3). It may be noted that one objective of this work is to re-use existing FORTRAN77 simulation code rather than translate it into C. For this purpose a C/FORTRAN77 interface was developed.

Current parameters are transferred as arguments to the remote C routine

using the UDP transfer mechanism. This made it necessary to develop XDR (EXternal Data Representation) routines which transform data particular to a specific machine into a machine-independent format. A limitation of 8 Kbytes exists which may in the future make it necessary to use other transfer mechanisms such as TCP which has no limit on the amount of data sent.

3.4.2 Signal handling routines

Initialization (reading all the default parameters) is done via an RPC call which will in principle have a successful return; sending the simulation parameters to the simulation program is done in the same way. However, execution time of the simulation program can be quite long and is usually unpredictable. In order to avoid blocking the interface until the remote routine returns, a short timeout for the RPC call is specified using the lower layer of the RPC package. An unsuccessful return occurs and is ignored, while execution continues in the remote machine. To compensate for the loss of the initial client/server link, flags are written to a log file whenever a change occurs in the execution status of the simulation program. This change can be the availability of new results, in which case writing the flag is straightforward, or events such as abnormal termination (e.g., due to arithmetic exceptions) of the simulation programs. This led to the need for the installation of a series of system signal handling routines, which catch signals generated by the system and write the appropriate flag to the log file. The interface possesses a routine that periodically checks the log file and takes apropriate actions according to the flag value, such as displaying a dialog box to alert the user to the crash of the simulation programs. This methodology makes it possible to know what is happening to the simulation programs running in background while the interface remains free for the analysis of results produced so far. System signals can be generated either by the system - and intercepted and handled by the signal handling routines - or by the user (e.g., to stop, pause or resume the integration) and sent to the remote program. A diagram summarizing the GUI/remote simulation program communication types is presented in Fig.4.

A possible installation problem resides in the fact that the C/FORTRAN77 interface (the C routines that "wrap" the FORTRAN77 routines) is compiler-dependent. If the simulation programs are run on another machine, it may be necessary to alter the source code of these routines.

3.5 Easy access to integration data

Data panels relative to the integration process only have been provided. The items consist of all the parameters needed by the LSODI integrator (Hindmarsh,1980). In case another integrator is used, the corresponding data panel must be changed as well.

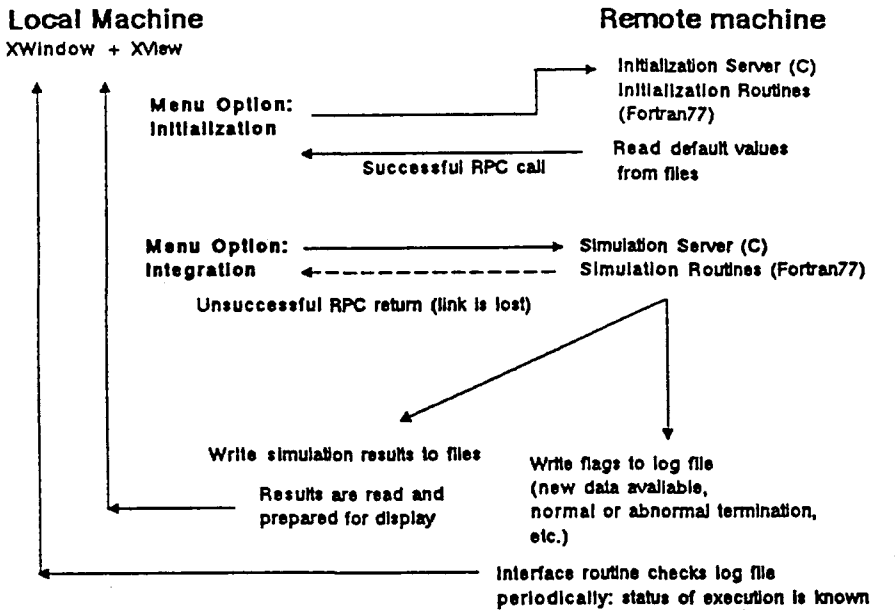


Fig. 4: GUI/simulation programs communication types.

3.6 Implementation of an effective on-line help system

Any kind of help can bring a dramatic improvement in user performance. Allowing users to control when to initiate a help request is better in terms of speed of task and error frequency (Rubin,1988). Context-sensitive help (i.e., help information that is related to the particular point the user has reached) has been selected. Although it is quite simple to create a help button in every window, on pressing which a text read-only window is displayed, the optimal structure of such a help system still has to be seriously thought through. For example, where should help pertaining to transitory pull-down menus be placed? Should it be a layered system - that is, more than one level of help? The final structure of the help system is still under study.

3.7 Error warning and recovery

In the case that errors are made by the user on the interface side, error detection is simple and performed by checking the value returned by the function. If an error value is returned, a dialog box is displayed.

In the case that errors occur on the simulation side, the strategy has been described above. Dialog boxes are displayed in any case.

3.8 Possibility of executing a selected set of UNIX commands through menu options

Including some UNIX commands as menu options was thought to be a convenient feature. The aim was to provide a very simple interface to the operating system in order to perform operations such as listing files or checking the processes currently being executed. All the already referenced commands to be sent to remote programs such as halting execution or resuming it are UNIX commands as well, in the form of system calls. Last, if the user wants direct access to the operating system, a terminal emulator window can also be created by the interface. This can be useful if a more experienced user wants to have full and unrestricted access to the operating system. The keyboard will then of course be used to input commands as in any command tool.

4 Conclusions

XView. together with the X Window libraries have been shown to provide a wide range of graphics capabilities and a useful set of general purpose widgets. XProcSim is a GUI for the graphical representation and management of data pertaining to a specific set of simulation programs. Its future development is to create a front end for process flowsheeting and/or dynamic simulation programs in general. Although out of the scope of the present work, selecting icons by the cursor and moving them to the desired location in a "drag and drop" operation would be an easy way to build the process diagram.

The general guideline, as far as GUI's are concerned, seems to be that there are no strict guidelines. It is however generally accepted that, to be fully effective, a GUI must be consistent, easy to learn by the novice user, provide shortcuts for the experienced user, obey ergonomic and human factors (such as adequate width and depth of menus) and even aesthetic considerations, etc. ... At this point of our work, it is obvious that user feedback is deeply needed in order to refine the prototype. This kind of practical information is probably, apart from theoretical considerations, the ultimate assessment of interface utility.

Acknowlegments. We are grateful to the Commission of the European Communities, under the Science Program grant contract No.B-SC1/900596, for the finantial support that made this work possible.

References

Drury, H. A., Clark, K. W., Hermes, R. E., Feser, J. M., Thomas Jr., L. J., Donis-Keller, H. (1992) A graphical user interface for quantitative imaging and analysis of electrophoretic gels and autoradiograms, BioTechniques, **12**, 892

Encarnação, J., Cote-Muñoz, J., Eckardt, D. (1991) User interfaces to support the design process, Computers in Industry, **17**, 317

Hindmarsh, A. C. (1980) Lsode and Lsodi: two new initial value ordinary differential equation solvers, ACM Signum Newletters, **15**, 10

Jensen, D. L., King, A. J. (1992) Frontier: a graphical interface for portfolio optimization in a piecewise linear-quadratic risk framework, IBM Systems Journal, **31**, 62

Kantorowitz, E., Sudarsky, O. (1989) The adaptable user interface, Communications of the ACM, **32**, 1352

King, G. B., Horlick, G. (1992) An advanced user interface for processing and displaying interferometric and spectral data: Spectroplot, Spectrochimica Acta, **47B**, E353

Miller, J. D. (1990) An Open Look at UNIX, M & T Publising, Inc., and Prentice-Hall International (UK) Limited

Moore, D. (1990) The migration of the X Window system, BYTE IBM Special Edition, Fall, 183

Nicholls, B. (1990) Looking at the graphical user interface, BYTE IBM Special Edition, Fall, 161

Nielson, G. M. (1991) Visualization in scientific and engineering computation, Computer, September, 58

Pangalos, G. J. (1992) Standardization of the user interface, Computer Standards & Interfaces, **14**, 223

Peddie, J. (1992) Graphical user interfaces and graphics standards, McGraw-Hill Inc.

Reiss, L., Radin, J. (1992) X Window inside & out, Osborne McGraw-Hill
Rimmer, S. (1992) Graphical user interface programming, Windcrest/McGraw-Hill
Rubin, T. (1988) User interface design for computer systems, Ellis Horwood Limited
Scheifler, R. W., Gettys, J. (1986) The X Window System, ACM Trans. Graphics, 5, 79
Schewk, C. F., Leaver, E. W., Schindler, H. E. (1991) The roles for process system simulation in the design, testing and operation of processes, process control and advanced control systems, J. Pulp and Paper Canada, 92, 53
Shandle, J. (1990) Real-time simulation speeds HDTV designs, Electronics, March, 68
Tufte, E. (1992) The user interface: the point of competition, Bulletin of the American Society for Information Science, June/July, 15

A METHOD OF EVALUATING THE USABILITY OF A PROTOTYPE USER INTERFACE FOR CBT COURSEWARE

Dr. Garry Patterson.

Dept. of information Systems, The University of Ulster, Jordanstown, Co. Antrim. BT37 0QB. Northern Ireland.

Abstract

This paper describes the evaluation process for assessing the usability of a prototype user interface for Computer-Based training (CBT) courseware. The MIDAS (Multimedia Interactive Design Aided System) model presents a methodology for the design, implementation and evaluation of multimedia CBT courseware. Results of the usability of the prototype application are presented which are based upon the evaluation criteria within the model. The evaluation results are analysed to test the performance of the prototype and their implications discussed. The implementation of the model presented is centered on the following application - *the training of pilots and engineers on the braking system of the British Aerospace Jetstream Aircraft.*

Keywords

Human-Computer Interface, Computer Based Training, Courseware, Multimedia, Evaluating Usability, Authoring.

Introduction

The research project was conducted under the general perspective of the definition of methods and tools for the evaluation of an interface based on human factors criteria. The goal was to design an interface for use by CBT courseware users in an effective learning environment. This study presents one such method; the validation of human factors criteria proposed within the MIDAS framework.

There are numerous different human factors methods currently available for the evaluation of interfaces. They each have advantages and drawbacks, the major problem however is that none of them allows a complete evaluation of an interface. Several classifications of such methods have been described ([Christie'90], [Karat'88], [Maguire'89] and [Senach'90]). Three main categories of methods can be distinguished depending on their orientation; theoretical models, expert judgement or the users. The latter approach is the one that will be stressed and used in this paper.

1 Evaluation Methods based on Users

In this category, evaluation methods are based on various user related data. The data may consist of users' performance data collected during user interaction with existing or prototyped interfaces (behavioural data) or from cognitive indications (understanding and memory). The data may also consist of subjective information extracted from users' attitudes and opinions about the interface. Extracting and monitoring user data is performed with several tools that are more or less intrusive. Such techniques include on site observation, audio-video recording, physiological recording or concurrent verbalisations. These can influence users' behaviour. Conversely, questionnaires, subsequent verbalizations, individual or group interviews have less of an influence on users' performance. The use of an *Evaluation Booklet*, presented the user with a practical method in the form of a checklist. The method adopted is discussed in Sections 2 & 3.

2 The Evaluation within the methodology of MIDAS

2.1 Overview of the Method

The method is a practical tool, in the form of a checklist. It is based on a set of software ergonomics criteria, which a well-designed user interface should aim to meet. The checklist consists of sets of specific questions aimed at assessing usability. These provide a standardized and systematic means of enabling those evaluating a CBT system interface.

2.2. The Evaluation Checklist

Each of the first ten sections is based on a criterion, which a well-designed user interface should aim to meet. The criterion is described at the beginning of the section, and a number of checklist questions follow, which aim to identify whether the interface meets the criterion. The criteria are not in any order of importance within the checklist. The ten criteria are summarised in Table 2.1.

When answering the checklist question, the evaluator of the CBT courseware ticks one of four alternative options. Of these, 'always' is the most favourable reply, and 'never' is the least favourable. The other two options are 'most of the time' and 'some of the time'. Research ([Hoinville'80], [Oppenheim'79]) has shown, that if a 'neutral' category was included as a fifth option, many subjects simply tick this option, whereas with the structure presented the subject has to think more carefully of the relevance of the answer to each *specific* question. There is space beside each question for the evaluator to make and comments of relevance to the question and to their answer.

At the end of each section, space is provided for the evaluator to add any other comments, good or bad, of relevance to the issues raised within the section. This is followed by a five-point rating scale, ranging from 'very satisfactory' to 'very unsatisfactory', which enables the evaluator to give a general assessment of the interface in terms of the criterion and questions within the section.

Table 2.1 Evaluation Criteria

Section Heading	Criteria Statement *(Summary)*
Visual Clarity	Information displayed on any CBT courseware screen should be clear, well-organized, unaambiguous and easy to read
Auditory Presentation	The music or voice commentary for any CBT screen should be clear, structured, unambiguous and easy to follow
Consistency	The way the CBT courseware looks and works should be consistent at all times
Compatibility	The way the CBT courseware looks and works should compatible with user conventions and expectations
Informative Feedback	Users of the CBT courseware should be given clear, informative feedback on where they are in the training course, what actions they have taken and whether they have been successful
Explicitness	The way the CBT courseware works and is structured should be clear to the user
Appropriate Functionality	The CBT courseware should meet the needs and requirements of users when carrying out tasks
Flexibility and Control	The CBT courseware interface should be flexible in structure, in the way information is presented and in terms of what the user can do, and to allow them to feel in control of the system
Error Prevention & Correction	The CBT courseware should be designed to minimize the possibility of user errors; users should be able to check their inputs and to correct errors
User Guidance & Support	Informative, easy-to-use and relevant guidance and support should be provided in the CBT courseware

Two further criteria are included, which relate to the general usability of the CBT courseware. Section 11 is specifically concerned with usability problems that the evaluator encountered when carrying out the tasks. For each question, the evaluator indicates whether there were major, minor or no problems. Section 12 consists of general questions on system usability. The questions are 'open', allowing the evaluator to express opinions of various factors, including the best and worst aspects of the CBT courseware interface system.

2.3 Sources of the Checklist Questions

The checklists have been developed with reference to a number of sources. A number of questions from the 'Usability: Screening Questionnaire' in ([Clegg'88]), and the format for these sections have been adapted from this source. In addition, early versions of Clegg provided the stimulus for a number of questions within Sections 1 and 10. Other sources for the checklist include ([Ravden'89], [Smith'86], [Gardner'87] and [Shneiderman'87]).

2.4 How should the Checklist be Used?

A key feature of the method, and a prerequisite for using the checklist, is that the evaluator uses the system tasks which the CBT courseware has been designed to perform, as part of the evaluation. This takes place before completing the checklist, and represents the first step in the evaluation. Tasks used in the evaluation should be representative of the work which is to be carried out using the system, and should test as much of the system, and as many of its functions, as possible. Such tasks require careful construction and are extremely important to the validity of the user interface evaluation. There are a number of reasons for use of representative tasks:

[i] Tasks which are realistic and representative of the work for which the system has been designed provide the most effective way of demonstrating the system's functionality.

[ii] This approach enables those evaluating the interface to see it not simply as a series of screens and actions, but as part of the application system as a whole.

[iii] By carrying out tasks, evaluator's can be exposed to as many aspects of the user interface as possible, which is necessary if they are to comment usefully, and in detail, on specific features, problems, strengths and deficiencies of the CBT courseware interface.

[iv] Many significant problems and difficulties of the courseware are only revealed when carrying out tasks.

[v] In some cases, there may be important aspects of usability which can only be captured by using the system (e.g. inflexibility of a menu structure when the task requires rapid movement between different parts of the system.)

In addition to the above points, important information can be gathered *by observing* an evaluator's performance when carrying out tasks in an evaluation. Observations and recording of task performance is extremely valuable in identifying those difficulties which evaluators experience when interacting with the system. As a supplement to the completed checklists, this information can enable, for example, investigation of critical confusions, common mistakes within the CBT interface. Examination of differences between evaluator responses to checklist questions, by referring to the task data; examination of task data for supporting evidence of an evaluator's comments about the interface; and comparison of evaluator groups according to their task performance are also issues which can be drawn from the learning environment.

2.5 Who should be involved in the Evaluation Process?

The method enables a variety of different people, with different background and expertise, to evaluate the same user interface. A particular benefit of this is that it enables end-users, such as the CBT students and trainers and designers who will use the system to access its usability before it is implemented into a full courseware package.

The degree of end-user involvement which is possible will depend, to a large extent, on the context of the evaluation. Factors such as the general availability of end-users, the scale and budget for the evaluation and the degree of novelty and complexity of the application may all influence the scope of user involvement. In addition to representative end-users and the CBT courseware designers, other evaluators such as software engineers not involved in the design and development of the interface being evaluated, who can provide an objective viewpoint from a more technical perspective; or cognitive psychologists and those in human factor groups, who can provide knowledge and understanding of the potential psychological implications of different designs.

2.6 When can the Method be Used?

The method can be used to evaluate usability both during and after design and development of the user interface. Where *prototyping tools* are available, it can enable early evaluation of an interface, or if alternative interfaces, so providing feedback at early stages in the design process. In this way, different configurations can be tested, and necessary improvements made, before reaching a fully operational state ready for implementation. At the later stages of development, it is more difficult to make major modifications to the interface.

The method can be of particular value in the later stages of development, where the

system is sufficiently developed to enable realistic tasks to be carried out. Similarly it can be used to evaluate a system before full implementation. This may be particularly valuable to an industry who intend to purchase from courseware developers that requires amendment, modification, or improvement before implementation.

The method can also be used in evaluating an application package with a view to purchase. Here tasks can help to assess whether the system meets the required functionality, and the evaluation can highlight good and bad features, aspects which are unsatisfactory, and those which are excellent. The method can also be used to evaluate alternatives for purchase, as it can allow a comparison between them.

In certain situations the checklist may not be sufficient to cover all the important usability issues which should be assessed in a particular evaluation. For example, where an interface relies upon speech input, some checklist questions will not be entirely appropriate and others may be added. The same may be true for systems which do not employ mainstream, relatively conventional user interfaces. It may be possible to augment or 'tailor' the checklist in these cases. Great care must be taken if augmenting the checklist, in order to preserve its balance.

2.7 Potential Financial Benefits of Using the Method

CBT user interface evaluation using this method may incur some immediate financial costs; for example, where the development process is lengthened, or through the involvement of end-users and others in the evaluation.

However, the importance of user interface design and of usability should not be underestimated, and its evaluation should be expected to take the time and effort worthy of its importance. In the longer term, the benefits which are likely to result from the evaluation will outweigh the costs for CBT courseware developers and companies which use their skills. Benefits may include:

- reduced training time for end-users;
- reduced support costs, due to fewer, and less significant difficulties;
- reduced need for amendments, modifications and revisions after implementation;
- where relevant, increased sales, as a more usable, well-designed and acceptable training environment is provided;
- a greater willingness among end-users to accept the training system and to use it effectively;
- a greater awareness among those developing CBT training courseware application systems of the requirements for 'user-centred' design.

In Section 3 the evaluation method is used to assess the usability of the prototype application user interface. In the light of experimentation, recommendations for

changes to the prototype are made. Section 4 presents the results from the Evaluation Procedure.

3 The Evaluation

3.1 Goal of the Evaluation

The goal is to assess the usability of a user interface for CBT courseware. The method, and in particular the checklist, enables an interface to be evaluated by a variety of people with differing expertise and backgrounds; including, for example, interface designers, other technical experts and, most importantly, representative end-users, who may or will actually use the system in practice. The method does not aim to solve problems, or to enable a quantitative assessment of usability. It provides a means of identifying problem areas, and the extracting of information concerning problems, difficulties, weaknesses and areas for improvement.

3.2 Method

3.2.1 Subjects

One hundred and twenty subjects participated in the study. There were 15 pilots whose experience in flying ranged from two to twenty-four years (M = 7.5; *S.D.* = 6.5), and the remaining participants were the student and staff population registered with The University of Ulster. Male and female subjects participated. The ages of the subjects ranged from 18 to 40 years (M = 23.7, S.D. = 4.7).

3.2.2 Equipment

The equipment used in the evaluation included a list of Evaluation Criteria that were designed to assess the usability of the user interface for the CBT courseware. Section 2.1 presented an overview of the method, and Table 2.1 summarised the criteria. The CBT courseware delivery platform was an Apple Macintosh IIfx with 12" high resolution colour monitor. Interaction was via a keyboard and mouse.

3.2.3 Procedure

Each subject participated in one individual evaluation session. The evaluation consisted of a learning phase and identification tasks in interactive mode. The subjects then completed the Evaluation Booklet. In the learning phase subjects were invited to study the prototype application relating to The Braking *System of the British Aerospace Jetstream 32 Aircraft*. They were encouraged to work through the introduction, statement of objectives, operation of the brake system and component location modules. However, the subjects could at any stage return to the main menu screen, and execute the courseware in any order. This would also assess the systems' usability. The identification tasks followed the completion of the learning phase. During these tasks, subjects were presented with problems relating to answering questions on the location

and operation of the braking system. No time limit was imposed on either phase. Times ranged from 25 - 40 mins. (M = 32.5, *S.D.* = 6.5) for the pilots; 29 - 72 mins. (*M* = 50.5, *S.D.* 13.6) for the engineering students; and 40 - 93 mins. (*M* = 65.0, *S.D.* = 11.9) for the rest of the subjects. Following the CBT lesson the subjects were required to fill in the Evaluation Booklet. This process took approximately 60 mins. to complete. The subjects were encouraged to fill this in immediately after the hands-on session, and to be as critical of the user interface (if necessary.) Spaces for written comments were provided beside each checklist question.

3.2.4 Data Collection and Analysis

Two types of data were collected. Firstly, the grading of the checklist questions, which was in the form of a [] in the appropriate box which best describes the subject's answer to each question, or **'N/A'** if the question was not appropriate to the courseware. Secondly, the subject may comment on any issue relating to each specific question in the appropriate column.

The objectives of the data analysis were threefold:-

Firstly, to assess the usability of the prototype application. To achieve this, global scores were calculated for each criteria question from the sample responses. A correct response was defined as either a [], **'N/A'**, or **'don't know'** in one of the appropriate columns on the checklist grid. If a checklist question was left blank, or two responses given, this was defined as a missing value. In the data analysis using SPSS, these values were eliminated, thus reducing the sample size.

Secondly, to (a) identify problem areas and weaknesses within the prototype application user interface; and (b) make recommendations to enhance a multimedia UID for CBT courseware.

Thirdly, using a factor analysis statistical technique, to identify key variables and subsequently design an Evaluation Booklet to assist CBT courseware designers in assessing usability of a UID for their end-users at the prototype stage of development. An explanation of the analytical technique - Principal Components Analysis is contained in Section 3.2.5

3.2.5 Analytical Technique - Principal Components Analysis

Factor techniques aim to produce new dimensions based on combinations of correlated variables, which clear the analysis of confusing multicollinearity ([SPSS'90], [Nie'75], [Johnston'78]). Given a set of n related variables, factor techniques transform *n* variable co-ordinate space into new *m* variable co-ordinate space where the new *m* variables (which are called '*factors*' or '*components*', depending on the particular form of the analysis used) are linear functions of the original *n* variables, which are now zero-correlated among themselves, that is, orthogonal. These new linear functions can

be powerful predictors of previously hidden relationships among the original variables. Common Factor Analysis (CFA) and Principal Component Analysis (PCA) differ essentially in the assumptions they make about the data, and therefore in the interpretations they permit. CFA is concerned only with the analysis of that portion of the total variance that correlates with other variables - the 'common reliable variance'([SPSS'90]). The common reliable variance, however, has to be estimated from that data, which means that CFA is part of inferential statistics, and must therefore make rigid data assumptions ([Maxwell'79], [Clark'73]). In contrast, *PCA is simply a mathematical procedure for variance manipulation, and as such makes no assumptions about the distribution of the data and no attempt to assess statistical significance. Moreover, in PCA the total variance is analysed, and the components may be thought of as simply a parsimonious description of the data.* The data obtained from the PCA technique in this study will be used to identify key variables for the design of the revised evaluation booklet and is obtained from executing the statistical package SPSS 4.0 on the Apple Macintosh computer environment.

To assist in arriving at an acceptable number of factors for the representation of the data, it is helpful to examine the percentage of total variance explained by each. The total variance is the sum of the variance of each variable. For simplicity, all variables and factors are expressed in standardized form, with a mean of 0 and a standard deviation of 1. In section one of the evaluation booklet, Visual Clarity, there are 14 variables, vc1, vc2,...,vc14 and each is standardized to have a variance of 1, the total variance is 14 in this example. The total variance explained by each factor is listed in the column labelled Eigenvalue. The next column contains the percentage of the total variance attributable to each factor. For example, the linear combination formed by Factor 2 has a variance of 1.43, which is 10.2% of the total variance of 14. The last column, the cumulative percentage, indicates the percentage of variance attributable to that factor and those that precede it in the table. Note, that although variable names and factors are displayed on the same line, there is no correspondence between the lines in the two halves of the table. The first two columns provide information about the individual variables, while the last four columns describe the factors. The table shows that 55% of the total variance is attributable to the first four factors. The remaining nine factors together account for only 45% of the variance. Thus, a model with four factors may be adequate to represent the data.

Several procedures have been proposed for determining the number of factors to use in a model. One criterion suggests that only factors that account for variance greater than one (the eigenvalue is greater than one) should be included. Factors with a variance less than one are no better than a single variable, since each variable has a variance of 1 ([Tucker'69]).

4 Results from the Evaluation Procedure

4.1 The User Interface of the Prototype Application

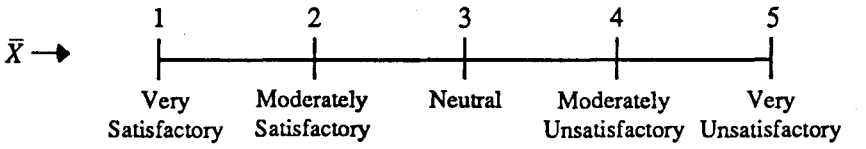
Table 4.1 provides a summary of the overall ratings for Sections 1 to 10 of the Evaluation Booklet.

These global scores provide a good indication of the general performance of the user interface for the prototype application. The results lead us to conclude that the design of the interface for the Braking System module provided a very satisfactory environment for learning. In five of the ten sections the range of values were between ($1.0 \leq x < 2.0$) and the other five between ($2.0 \leq x < 3.0$). The overall mean for the ten sections being

Table 4.1 Summary of Overall Ratings for the Evaluation Criteria

Criteria Headings	*N*	*MV*	$\overline{X}$	*S.D.*
Section 1 - Visual Clarity	120	0	1.55	0.71
Section 2 - Auditory Presentation	120	0	1.77	0.88
Section 3 - Consistency	118	2	1.61	0.71
Section 4 - Compatibilty	118	2	1.81	0.76
Section 5 - Informative Feedback	120	0	2.15	0.95
Section 6 - Explicitness	118	2	2.01	0.81
Section 7 - Appropriate Functionalilty	119	1	1.95	0.82
Section 8 - Flexibility and Control	116	4	2.41	0.93
Section 9 - Error Prevention & Correction	112	8	2.25	0.94
Section 10 - User Guidance & Support	85	35	2.67	1.33

Rating Scale



The lowest rating scores were identified in Section 10. In this category, User Guidance and Support, only 85 subjects completed this section. The reasons for this can be identified by examining the questions presented in the Evaluation Booklet Section 10.

The only user support provided in the courseware was audio and visual feedback if incorrect answers were given by the subjects to the questions. There was *no* Help menu integrated into the braking system to provide on-line assistance for the aircraft terminology or hardcopy output to supplement the CBT material. The subjects did not have a copy of the training manual to complement the CBT instruction. The data from this section supports these claims. The booklet used in the evaluation process is a generic document for UID evaluation and thus by including options which were not included in the prototype design, can be remedied at the implementation stage of development.

The use of a multimedia platform proved a very successful user environment for learning. 63% of the subjects commented on the effective use of multimedia to present the training material, with a further 10% supporting the specific use of graphics and animation. The lack of an on-line Help facility is also noted as one of the worst features of the system. 33% included comments, specifically mentioning the lack of feedback from the system. A further 24% commented on a lack of explanation of the direction icons. In the written comments, the subjects however noted that once they became familiar with their functionality, they did not pose a problem. It is very encouraging to note that in general terms some 63.5% of all subjects had no difficulty in either understanding the system or in finding any irritating system problems. In recommendations suggested by the subjects for improvement, only 23.3% included the inclusion of a Help facility. Over 55% concluded that the CBT courseware represented a good prototype application. A further 10% suggested that the provision of moving-video illustrations would have enhanced the learning environment.

No evidence existed to suggest that the different subject groups found the user interface environment more complex to use or the training material any more difficult to understand. One reason for this is that no assumed knowledge was required for an understanding of the braking system. The pilots commented on the effective use of multimedia in designing a realistic training environment. In the past, this had only been available when the pilots progressed from the classroom instruction to the Flight Training Devices, and even then this was a static environment. Only when the pilots progressed to training in the Full Flight Simulator was interactivity a part of the learning experience.

4.2 The Revised Evaluation Booklet

The Revised Evaluation Booklet has been designed to assist designers and end-users in assessing the usability of a UID at the prototype stage of development. The questions included in each section are identified from the analysis of performing PCA on the subjects data. The coefficients (factor loadings) in the Factor 1 column of the FACTOR MATRIX. Factors with large coefficients ($0.55<x<1$), (in absolute value) have been used in rank descending order to indicate the questions to be included in the reconstruction of each sections' questions.

5 Conclusion

The objectives of this paper were to assess the usability of a user interface for multimedia CBT courseware and to design an abbreviated evaluation booklet to be used in training environments. An evaluation booklet developed within the MIDAS framework was used for this purpose. The data analysed provided overwhelming support for the use of multimedia as the delivery platform for presenting CBT courseware. The data also identified areas of weakness and problems within the UID. Within the prototype application reviewed by the subjects, the absence of an on-line Help facility was identified. PCA was used to identify the key variables within each evaluation criterion. The revised evaluation booklet designed for use by designers and end-users of multimedia CBT courseware in training environments is discussed. The data analysed provides overwhelming support for the use of multimedia as a delivery platform for presenting CBT courseware.

References

[Christie'90]: Christie, B and Gardiner, M M: "Evaluation of the human-computer interface", *Evaluation of human work: a practical ergonomics methodology*, pp. 271-320, Taylor and Francis, 1990

[Clark'73]: Clark, D: "Normality, Transformation and the Principal Components Solution: An Empirical Note", *Area*, Vol.3, pp. 112-8, 1073

[Clegg'88]: Clegg, C W; Warr, P B and Green, T R G *et al*: *People and computers - how to evaluate your company's new technology*, Ellis Horwood, Chichester, 1988

[Gardner'87]: Gardner, M M and Christie, B (Editors): *Applying cognitive psychology to user-interface design*, Wiley & Sons, Chichester, 1987

[Hoinville'80]: Hoinville, G; Jowell, R and Airey, C *et al*: *Survey Research Practice*, Heinemann Educational Books, London, 1980

[Johnston'78]: Johnston, R J: *Multivariate Statistical Analysis in Geography*, Longman, London, 1978

[Karat'88]: Karat,J: "Software evaluation methodologies", in *Handbook of Human-Computer Interaction*, edited by M Helander, Elsevier Science (North Holland), Amsterdam, 1988

[Maguire'89]: Maguire, M and Sweeney, M: "System monitoring: garbage generator or basis for comprehensive evaluation system?" in *People and Computer V*, edited by A Sutcliffe and L Macaulay, Cambridge University Press, Cambridge, 1989

[Maxwell'79]: Maxwell, A E: *Multivariate Analysis in Behavioural Research*, Chapman and Hall, London, 1979

[Nie'75]: Nie, N H; Hull, C H and Jenkins, J G *et al*: *SPSS: Statistical Package for the Social Sciences (2nd Edn.)*, McGraw-Hill, New York, 1975

[Oppenheim'79]: Oppenheim, A N; *Questionaire Design and Attitude Measurement*, Heinemann, London, 1979

[Ravden'89]: Ravden, S and Johnson, G: *Evaluating Usability of Human-Computer Interfaces - A Practical Method*, Wiley & Sons, Chichester, 1989

[Senach'90]: Senach, B: *Evaluation ergonomique des interfaces homme-machine: une revue de la littérature* (English translation), Institut National de Recherche en Informatique et an Automatique, Rocquencourt, France, 1990

[Shneiderman'87]: Shneiderman, B: *Designing the User Interface: Strategies for Effective Human-Computer Interaction*, Addison-Wesley, Reading, MA, 1987

[Smith'86]: Smith, S L and Mosier, J N: *Guidelines for designing user interface software*, Report No. MTR-100090; ESD-TR-86-278, The Mitre Corporation, Bedford, MA, 1986

[SPSS'90]: SPSS Inc., *SPSS Base System User's Guide*, SPSS, Chicago, 1990

[Tucker'69]: Tucker, R F; Koopman, R F and Linn, R L: "Relations of factor score estimates to their use", *Psychometrika*, 36,pp. 427-436, 1969

Modeling the Answering Partner of the Dialogue Process

Oleg P. Pilipenko

Odessa University
2 Petra Velikogo St., Odessa 270057, Ukraine, CIS

1 Partners' dialogue

Functions of a dialogue system are problem-independent and are determined only by specifics of human-computer interaction. This fact allows us to separate user interface from application programs and to build a formal model for dialogue systems. Such a model can be used as a methodological base for CASE-tools to develop dialogue systems. This model of dialogue must be constructive: simple enough to be implemented and complex enough to cover a wide range of dialogue processes.

Some authors consider human-computer dialogue as a sequence of question-answer pairs. Participants' roles are strictly regulated - one of them is asking (active partner), another is answering (passive partner). However, our experience shows that this kind of dialogue is not widespread in everyday communication. It is found almost exclusively in formal teacher/student settings. More frequently initiative in dialogue is passed from one participant to another. We shall name such kind of dialogue *partners' dialogue*. The answering partner can intercept initiative when he does not understand a question, does not have enough information to answer a question, or an error occurred while answering a question. The answering partner may also intercept initiative when the asking partner fails to formulate a question correctly and the answering partner needs to ask questions to define asking partner's demands more precisely and to give an acceptable answer finally.

It is particularly important to be able to model the above mentioned partners' dialogue in order to develop modern dialogue systems. Partners' dialogue is more productive because it mimics our natural dialogue usage.

2 Automaton model

Let us define automaton P as quadruple $< S, D, E, h >$, where S is a set of automaton states, where the initial state s_0 and the final state s_n are

distinguished; D is a set of inputs; E is a set of outputs and $h : S \times D \to S \times E$ is a function that calculates automaton output e and next state $\bar{s}$ according to the current state s and automaton input d (user's action).

Let us consider automaton state classes. Automaton can:
- answer a user's question and wait for the next question;
- ask a question and wait for a user's action;
- answer a question, intercept initiative, ask user a question, and wait for user's action.

We shall designate them accordingly S^1, S^2 and S^3. The final state s_n remains unmapped. We can note that $S = S^1 \cup S^2 \cup S^3 \cup s_n$.

We shall introduce some sets: Q - set of user's questions, A - set of user's answers, Q' - set of automaton questions and A' - set of automaton answers. There are null answer a_0 in A'. For each s_k we shall define special sets: Q_k - set of discernible user's questions, and A_k - set of discernible user's answers. Let there are exist input d_0 which cannot be recognized as a discernible user's question or answer at any step: $d_0 \notin \bigcup_{k=1}^{n} Q_k$, $d_0 \notin \bigcup_{k=1}^{n} A_k$, and $d_0 \in D$. The automaton set of inputs consists of possible user questions and answers ($D = Q \cup A$), and its set of outputs contains it's own answers and questions, that it asks intercepting initiative in dialogue ($E = Q' \cup A'$).

Let us examine output function f. Assume that automaton is in the s_k state. If user's action d is recognized as question ($d \in Q_k$), then automaton can proceed by answering the question, or intercepting initiative and asking the user a question of it's own. If a user's action is recognized as an answer ($d \in A_k$), then automaton will ask the user a new question or answer the most recent unanswered question.

Let us examine transition function g. Assume that automaton is in the s_k state. If $s_k \in S^1$ and the user's action d is not recognized as a discernible question or answer, then the $g(s_k, d)$ state belongs to S^2. If $s_k \in S^1$ and the user's action is recognized as a question, then the $g(s_k, d)$ state belongs to any class of states (excluding final state s_n). If $s_k \in S^2 \cup S^3$ and the user's action is not recognized or recognized as a discernible answer, then the $g(s_k, d)$ state belongs to S^2. If $s_k \in S^2 \cup S^3$ and the user's action d is recognized as discernible question, then the $g(s_k, d)$ state belongs to any kind of states (including the final state s_n).

Let us consider the automaton cycle. If current automaton state $s_k \in S^2$, automaton will ask a question. If $s_k \in S^1 \cup S^3$, automaton will begin step with visualization of the answer which was obtained on the previous step. If

$s_k \in S^3$, after the answer is visualized automaton will intercept initiative and ask question $q \in Q'$ connected with the current state. Following any of the given situations, automaton will wait for the user's action. In any state automaton must recognize some of the user's actions as questions. The accepted user's action is either recognized as a question from Q_k or answer from A_k, or it remains unrecognized. According to accepted action automaton produces an output $e = f(s_k, d)$. If the action remained unrecognized, then $f(s_k, d) = f(s_k, d_0)$. Than automaton transits to the new state $g(s_k, d)$.

Automaton starts from the state s_0 . Automaton can start dialogue as an active partner - by asking the user a question or, as passive partner, waiting for the user's question. In the first case $s_0 \in S^2$. In the second case we consider that $s_0 \in S^1$ and that automaton will start this step with visualization of the null answer a_0. When automaton transits to the s_n state, it stops.

Automaton models are interesting because it is simple to implement such models on a variety of contemporary computers.

3 Net model

The state-transition diagram of the above mentioned automaton can be mapped to a network $G = < S, \Gamma >$ with the node set S and the relationship Γ that sets up a correspondence between node s_i and the set of nodes $\Gamma(s_i)$ which are at the ends of the edges starting from this node. All edges starting from each node are numbered. Let us change transition function g slightly in such way that $g : S \times D \rightarrow N$. Transition from the current node will occur to the node ending the edge with number $g(s_k, d)$. We shall not represent the final node s_n. The dialogue system will finish its work when $g(s_k, d) > card\ \Gamma(s_k)$, in other words, when $g(s_k, d)$ is greater than number of the edges starting from the current node.

We shall name the set of network and step function the *dialogue scenario.*

4 Net model implementation

Net model is useful as a formal basis for the universal dialogue interpreter which can conduct dialogue according to a stored dialogue scenario.

Function h can be implemented by the internal facilities of the dialogue interpreter or by use of *external processes.* Answer to the user's question and/or transition code can be calculated by a separate executable module invoked by system.

Database or knowledge base can be connected with such a dialogue system. It can contain some facts, data or rules on how to analyze the user's questions or how to synthesize answers. Then, some steps of dialogue can be connected with database (knowledge base) modifications.

In order to implement dialogue scenario generator based on above mentioned net model, we need to choose interpretation of sets S , Q and A. We may consider any key press, mouse click or mouse movement as a user's action. However, such interpretation would lead to an unnecessarily detailed dialogue scenario consisting of an unmanageably large number of nodes. To reduce complexity of the dialogue scenario networks we can use their recursiveness. We can replace some subgraphs with nodes and, in that way, increase the "granularity" of nodes and possible user questions and answers. We can substitute nodes for subgraphs satisfying the following condition: all edges entering subgraph enter the same node. We shall name such subgraphs substituted by nodes *scenario units.*

Usual user interface objects, such as entry fields, menus, dialogue boxes with pushbuttons, radio buttons and check boxes, help, may be considered as standard scenario units. Conception of the scenario units allows us to develop dialogue scenarios by means of incremental refinement. We can provide the reusability of scenario units: the same units can be used instead of the different nodes or even in the different dialogue scenarios.

The dialogue partner model was implemented in the Dialogue Processor project. The Dialogue Processor consists of a Dialogue Generator to generate dialogue scenarios, and a Dialogue Interpreter to interpret these scenarios. The Dialogue Generator stores a generated scenario as a metafile - a stream of objects. The scenario determines permissible transitions (net edges). Each dialogue step (net node) can include passive audio and video objects, active video objects and description of external process that the step invokes. Passive objects are used to express a question or an answer for the user's question. Active video objects, are used to give the user an opportunity to answer a question or to ask his own question.

Passive video objects are panels with static text or browsers for texts, tables or bit maps. Animated passive video objects and audio objects add an expressive force to generated dialogues. Active video objects are standard dialogue panels - menus, choice lists, or dialogue boxes with entry fields and choice fields of different kinds.

5 Summary

The suggested answering dialogue partner model is methodological base of the Dialogue Processor - the tool to generate dialogue systems. The Dialogue Processor uses object-oriented technology and stores dialogue scenarios as streams of objects. Further development of object-oriented

databases will allow storage in one database dialogue objects and application objects, and production of more complex dialogue systems.

The suggested model can be extended to describe dialogue partner training. If we assume that step function, sets of net nodes and edges, or sets of discerning questions and answers depends on time, we shall obtain model of a non-stationary dialogue system. If we add system actions that modify the above mentioned dialogue system elements, we shall get a model of dialogue partner that can be taught.

Intelligent Assistance for Simulator-Based Training

Eva L. Ragnemalm

Department of Computer Science, University of Linköping, S-581 83 Linköping, Sweden
E-mail: elu@ida.liu.se

This poster presents an ongoing project aiming at the design of a prototype system intended to provide intelligent assistance for simulator-based training of process-control operators in the domain of paper and pulp manufacture.

The purpose of the training in our case is to increase the operator's understanding of the process and to operationalize his knowledge in a diagnostic context. It is intended to supplement classroom instruction and on-the-job training.

One difference between skilled process operators and novice operators is the ability to diagnose errors efficiently. Skilled operators follow a diagnostic procedure that appears to be transferrable (Schafstaal 1991).

Teaching this diagnostic procedure to novice operators in order to operationalize their knowledge would involve presenting a fault situation on a simulator and following the operator through the steps of the procedure. The problem is that many of the steps are not normally verbalized or otherwise made explicit. We are considering getting around this by engaging the user in a dialogue using a Learning Companion (Chan, 1990).

This dialogue must be easy for the student to engage in, and must not interrupt his thinking in a distracting manner, since concentration is important when learning in a dynamic environment (Munro, Fehling and Towne, 1985).

One benefit of the learning companion as a dialogue technique would be the possibility of actively analysing the student's understanding of the system to be learned, while still preserving the exploratory nature of learning with a simulator.

A nice side-effect is that in forcing the student to express his reasoning, one increases the learning potential. The dialogue also makes the learning situation more similar to reality, in which diagnosis takes place as a cooperative effort among the operators in the control room.

DIALOGUE SPECIFICATION LANGUAGE AND TOOLS FOR AN INFORMATION SYSTEM OVER THE TELEPHONE

Andrés Santos, José Colás, Juan Lestani
Dpto. Ingeniería Electrónica - E.T.S.I.Telecomunicación
Univ. Politécnica Madrid - Spain

1. INTRODUCTION

Speech technology is already in the stage of producing applications. An important area is based on services to provide information through the telephone without intervention of a human operator (see figure 1). Today there are several companies that provide limited information using touch-tone telephones. These systems have at least two basic problems: the user can access to the information in a restricted and artificial way (menu-driven with codified commands) [1-3] and touch-tone telephones are not available everywhere (in some countries, like Spain, the percentage of touch-tone telephones is very small, around 5 %).

The success of automatic information systems depends highly on the use of speech technologies. Speech is the natural way to communicate among people of every culture. Any useful system needs speech recognition and speech production capabilities to operate through a telephone line. Above these modules there should be another one to control the dialogue flow so that the user gets the information: this module can be called the *dialogue manager* [4-6] (see figure 2). Among other functions, it controls the interactions between the system and the user: it receives requests, decides whether more specific data from the user are needed, gives the information if available, etc.. To perform these tasks, it also has to interact with the application, using some access functions.

Considering man-machine communication, there are two main restrictions: one is due to the capacities of the speech recognizer; the other to the possibilities of the system to *understand* and *generate* speech:

a) There are several kinds of recognizers depending on their vocabulary and complexity, from the simplest that can only recognize isolated digits. More complex systems would have possibilities to recognize continuous speech with large vocabularies. Obviously these differences have a big impact on the performance of the whole system [7, 8].

b) Depending on the system's ability to understand the user's speech, the dialogue can be menu-driven or unrestricted. In the first case, the user can

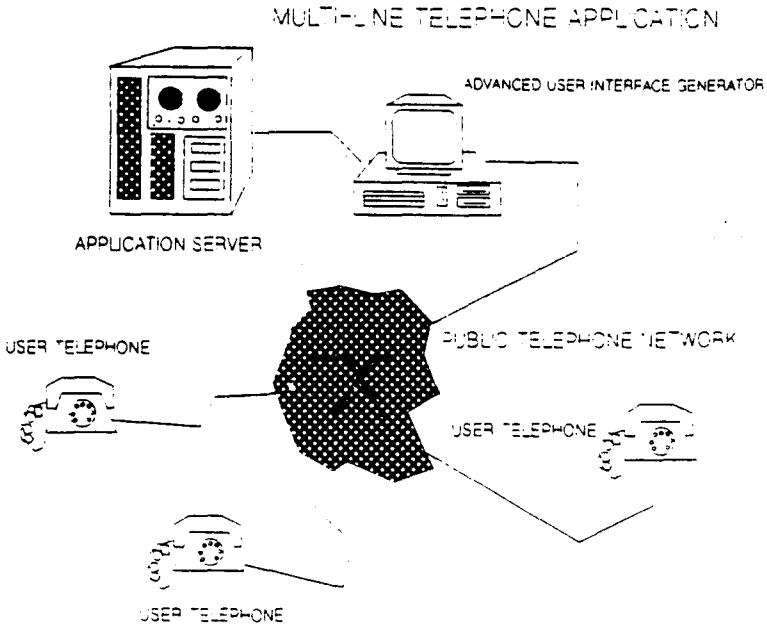


Figure 1: Basic structure of an information system over the telephone

only give answers included in a menu; the dialogue is then very limited as the user cannot give any unsolicited data. In the second one, the dialogue is much more flexible as the system can accept any answer; if it finds that more specifications are needed to provide the information solicited, it asks for such data [9, 10].

In this paper we will consider first different kinds of information systems with different complexities. In the following section the basic modules and tasks that are included in a general system will be presented. A high-level language to define the dialogue and the operation of the system is described next. Finally the application generator tool and its possibilities are considered.

2. SCENARIOS AND STRATEGIES OF DIALOGUE

Dialogue can be thought as a sequence of communicative acts between the user and the information system. The behaviour of this system should have the objective of providing the information seek by the user (it has to be *successful*), but two subojectives could be defined: the minimization of interactions of dialogue (*effectiveness*) and the similarity with the dialogue among humans (*naturalness*).

With these objectives in mind, different models of system behaviour, different scenarios, could be defined. In each scenario the user has different degrees of flexibility to convey information to the system. The vocabulary and the linguistic contents that can be used are defined. So are the quantity and the kind of information that are interchanged in each dialogue intervention.

The system should transmit to the user these restrictions imposed on his/her answer. They define a *dialogue strategy*.

Possible strategies are the following ones:

Menu-Driven:

a) Isolated Digits
b) Isolated Words
c) Digit spotting
d) Word spotting

System: Hello, this is Sports Information Service. Please, speak carefully, one word at a time. Is this the first time you use this system, yes or no?
User: No.
S: You said no. Please, say one if you want football, say two if your choice is basketball or say three if you want tennis.
U: Two.
S: You said two. Information about basketball. The last results in the European Cup are...

Figure 3: Example of dialogue using an isolated-digit recognizer

In the first two, the system has to make very restricted (and artificial) questions to constrict the user answer (see the example in figure 3). The third and forth allow in some way more natural questions, as the user answer is not so limited (figure 4).

Unrestricted Speech (Natural-Language):

e) Multiple keyword spotting (island-driven)

f) NL without dialogue history (semantically complete sentences)

g) NL with dialogue history (including sentences with ellipsis or references to previous questions).

In these systems the requirements imposed on the module that processes the user inputs are much bigger. The questions are much more natural and the human answer is not so restricted (example in figure 5).

3. BASIC ARCHITECTURE OF A DIALOGUE SYSTEM

The general structure of an information system is shown in figure 2. In the following paragraphs, its blocks will be commented [11, 12].

3.1. Input Manager

This module is part of the interface with the user. It provides the dialogue manager with the inputs from the user. Depending on the scenario, its tasks have different complexity and can be divided into two different modules:

a) **Speech recognizer**: this is the acoustic module that tries to recognize the speech produced by the user and to give the corresponding sequence of words (text). It uses a specific vocabulary (not being able to recognize any word not included in this vocabulary), which may be different on each phase of the dialogue.

The recognizer could make some *errors*: part of the input could be misrecognized.

b) **Linguistic module**: It works at the syntactic and semantic levels. Its objective is to produce a semantic representation of the user's input. It has linguistic information that could be a grammar, semantic rules, statistical data, etc.. In some systems, like in menu-driven dialogues, this module is very simple or does not exist at all.

System: Hello, this is the train information system. Please, answer the questions speaking carefully. At any point, you can say help to obtain more information about this system or operator to speak to a human operator.
S: Where do you want to leave from?
U: From Madrid.
S: Where do you want to go?
U: To Valencia.
S: At what time?
U: Well... around 3 p.m.
S: So, you want to go from Madrid to Valencia approximately at 3 in the afternoon. Is that right?
U: Yes.
S: There is a train at 3.15 ...

Figure 4: Example of dialogue using a word-spotting recognizer.

S: Hello, this is the automatic train information system.
U: I want some information about trains to Valencia.
S: From where do you want to go to Valencia?
U: From Barcelona.
S: At what time?
U: In the afternoon.
S: So, you want to go from Barcelona to Valencia in the afternoon. Is that right?
U: No, sorry. I want to go in the morning.
S: O.K. You want to go from Barcelona to Valencia in the morning. Is that right?
U: Yes. In the morning.
S: Well. You have an Intercity at 8 o'clock...

Figure 5: Example of unrestricted dialogue

Its *errors* are semantic: in the formalism used, it could not be possible to represent the input text or the representation found is not correct.

3.2. Output Manager

a) **Synthesizer**: The speech can be produced either with a text-to-speech converter (unrestricted vocabulary) or storing fixed messages. The first method gives much more flexibility to provide information, but its quality is usually lower.

b) **Message generator or selector**: Depending on the system used to produce speech, this module could just select one of the predefined messages, or it could produce the text to be sent to the synthesizer. Some techniques to generate natural language could be used in the most flexible scenario.

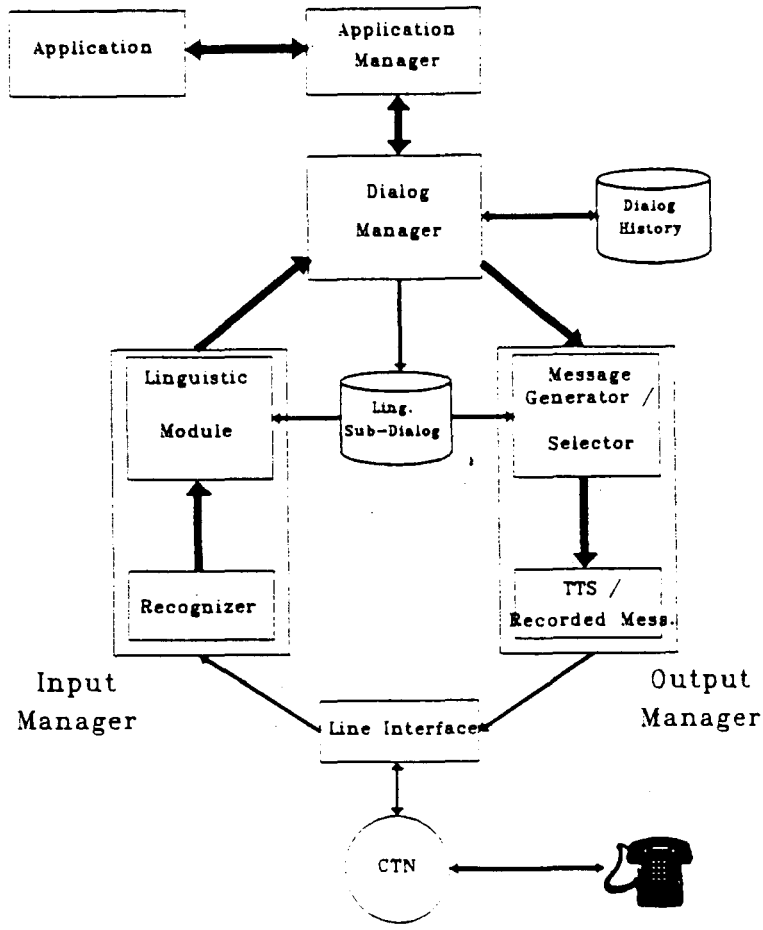


Figure 2: Modules of a general information system

The messages produced by the system can be classified in four groups:

- Information requests: to complete or to define the user's demand.
- Confirmation of inputs.
- Answers - Information requested.
- Error Messages.

3.3. Application Manager

This is the only application specific module [13]. It translates the *application functions* used by the Dialogue Manager into commands and functions needed by the application. This module allows the rest of the system to be independent of the specific application.

3.4. Dialogue Manager

This is the main module that guides the dialogue. It works at the semantic level and interacts with the input and output managers and with the application manager.

The dialogue manager can be considered a finite automata. The transition between states is determined by the input module. Each state can activate two kinds of functions:

- applications functions: generally queries of a data base; they interact with the application manager to obtain the requested information if available.

- internal functions: to maintain the history of the dialogue to solve further ellipsis and ambiguous queries. They also provide some help to the user, call a human operator, etc.

This finite automata can be studied in two different scenarios:

In Menu-driven applications the sequence of states that control the evolution of the dialogue seeks to obtain a complete query to access the data base. The user goes through a path of dialogue interactions until his/her request is fully defined.

In Natural-Language applications the dialogue manager builds a data structure that contains the information needed to produce a query in the formalism defined by the application manager. This dialogue manager should identify the missing information in the request made by the user

and made the corresponding question to obtain it.

In both cases, the dialogue manager has two additional tasks:

Error handling: the manager has to identify the errors that could be produced in the other modules and take the corresponding action (it usually generates an error message).

Dialogue history: it has to maintain an internal data structure to keep track of the valid data in the previous dialogue interactions. This structure will allow the completion of further questions.

A dialogue state can be defined. In menu-driven systems it will be the state of the automata. In natural-language applications it will be associated to the main data structure. In both cases, the dialogue state will include:

- a set of possible interactions with the application
- a linguistic sub-dialogue database: it could be the vocabulary available to the speech recognizer, the set of messages to be produced, etc.
- a set of possible errors and their handling procedures.
- a set of linguistic rules (grammar rules) to define the syntactic structure of the valid sentences in that point of the dialogue.

4. DIALOGUE DEFINITION LANGUAGE (DDL).

As seen in the previous section, the whole architecture to build an application is complex, specially if we want a flexible and in some way intelligent dialogue. To build commercially viable applications, we should provide the *application developer* [14] with a tool to customize a standard architecture. We expect the application developer to concentrate on designing a well structured dialogue, but he/she will not have a through knowledge of the internal operations of the system nor the speech synthesizer/recognizer.

Therefore, a high level language to describe the dialogue must be defined. It is called Dialogue Description Language (DDL) [15, 16] or Dialogue Specification Language (DSL).

DDL is a formal language to describe the operation of the whole system as a finite state machine (see an example in figure 6). To define an application using DDL, the following elements should be defined:

a) A set of dialogue states, each with a data structure and a history.

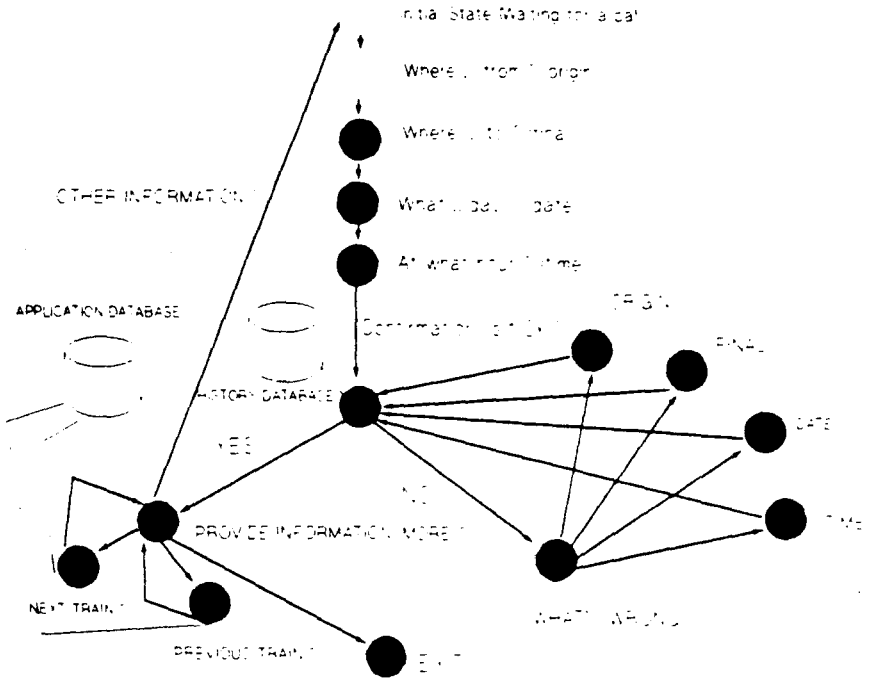


Figure 6: State diagram of a "train information system"

b) The transitions between states.
c) Functions, divided in three classes: I/O, application and control functions.
d) A subset of the language (vocabulary plus messages) active in each state.
e) A set of error handling procedures.

In the rest of the section we are going to describe our own DDL, oriented specially for menu-driven applications, although it can be easily expanded to unrestricted speech applications.

The following types of functions have been defined:

- I/O instructions:
  * Line Control: wait_for_call, answer_phone, hang up, ...
  * Speech output functions: synthesize, send_message (pre-recorded), record, ...

    * Speech input functions: recognize_vocabulary[1], recognize_digit_string, ...
    * Error handling: send_error_message, call_to_operator, ...

- Control instructions:
    * Flow control (transitions between states): goto, subroutine call, etc..
    * History keeping functions: store data, etc..

- Application Functions:
    * Search Functions

In the Annex an application described in DDL is show: the *train information system* whose block diagram was shown in figure 6.

5. A MENU-DRIVEN APPLICATION GENERATOR WITH A WIZARD-OF-OZ MODULE INCORPORATED

The *Application Generator* is a tool for the application designer that allows the definition of the communication with the user and with the application using a Dialogue Description Language.

The application generator is a shell of the system architecture. It customizes the architecture modules and the internal data structures.

The application generator should have the following functions:

* Editor (textual or graphical) to describe the application in DDL.

* Parser to detect errors in the description.

* Compiler to translate the application into the architecture, customizing the modules to implement the described functions.

* Help messages for the designer: about system characteristics, language functions and syntax, etc.

* Tools to edit oral messages, produce tones, cut & paste voice fragments, etc.

---

[1] The function that calls the recognizer deals with certain errors: no word recognized, timeout, too many errors, the user hangs up, etc.

* Application Simulator: The system can be debugged simulating the interactions with the application and the users. The speech recognition module can be replaced by a human operator that listen to the dialogue and simulates the recognizer. In that way, the system goes from one state to the other, following the designed automata. This method is known as the *wizard-of-oz* [6].

Figure 7 shows a block diagram of the whole system.

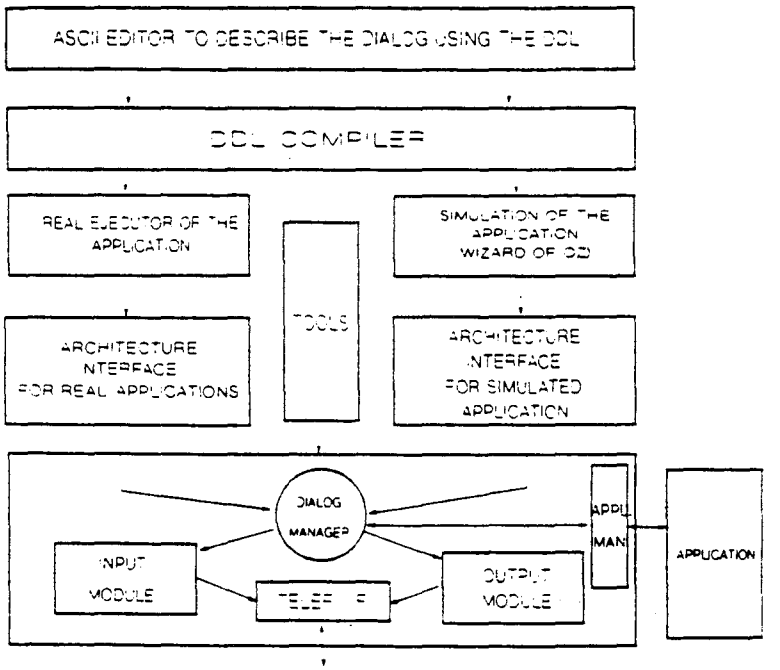


Figure 7: Block diagram of the application generator

6. CONCLUSIONS

A general model for automatic information systems has been presented. The structure has to be easy to customize and modify. So a Dialogue Definition Language (DDL) has been defined. It is fully adequate for menu-driven applications. For unrestricted speech, further research is needed. The general structure seems adequate, but there is not a general approach to translate the speech produced by the user into a data structure to guide the dialogue and the access to the application.

ANNEX: Application described in DDL

```
begin:
   wait_for_call;
   answer_phone;
   send_message saludo;

   Pedir_Origen;
   Pedir_Destino;
   Pedir_Fecha;
   Pedir_Hora;
Comprobacion:
   send_message Peticion_Completa;
   send_message Es_Correcto;
   recognizer
      Palabras_Activas = SN;
      Intentos = 2;
      case si :   Goto Buscar_Inf
      case no :  Goto Corregir
   end_recognizer;
Buscar_Inf:
   external_function Buscar_Tren;
   send_message Inf_Tren;
Sig_Tren:
   send_message SigAnt;
   recognizer
        Palabras_Activas = {Siguiente,Anterior,Otro,Repite};
        Intentos = 2;
        case siguiente :
            external_function Buscar_Tren_Siguiente;
            send_message Inf_Tren;
            internal_function Actualizar_HORA;
            send_message SigAnt;
            Reintentar;
        case anterior :
            external_function Buscar_Tren_Anterior;
            send_message Inf_Tren;
            internal_function Actualizar_HORA;
            send_message SigAnt;
            Reintentar;
        case otro :
            Goto Corregir;
        case repite :
            send_message Inf_Tren;
            send_message SigAnt;
            Reintentar;
        case nada :
            Goto despedida;
   end_recognizer;

Corregir:
   send_message Que_Corregir;
   recognizer
```

```
        Palabras_Activas = {Origen, Destino, Día, Hora};
        Intentos = 3;
        case origen :
            Pedir_Origen;
            Goto Comprobacion;
        case destino:
            Pedir_Destino;
            Goto Comprobacion;
        case dia:
            Pedir_Fecha;
            Goto Comprobacion;
        case hora:
            Pedir_Hora;
            Goto Comprobacion;
        case nada :
            Goto despedida;
    end_recognizer;

Pedir_ORIGEN:
    send_message diga_ORIGEN;
    recognizer
        Palabras_Activas = Ciudades;
        Intentos = 3;
    end_recognizer;
    internal_function Guardar_ORIGEN;
    return;

Pedir_DESTINO:
    send_message diga_DESTINO;
    recognizer
        Palabras_Activas = Ciudades;
        Intentos = 3;
    end_recognizer;
    internal_function Guardar_DESTINO;
    return;

Pedir_FECHA:
    send_message diga_FECHA;
    recognizer
        Palabras_Activas = Dias;
        Intentos = 3;
    end_recognizer;
    internal_function Guardar_FECHA;
    return;

Pedir_HORA:
    send_message diga_HORA;
    recognizer
        Palabras_Activas = Horas;
        Intentos = 3;
    end_recognizer;
    internal_function Guardar_HORA;
    return;
```

```
despedida:
   send_message despedida;
   hang_up;
   Goto inicio;
end:
```


Acknowledgements

The authors wish to thank Juan Manuel Montero for his suggestions to improve this paper and Francisco Javier de Pedro for his contributions to the definition of DDL. This project has been supported by Comunidad de Madrid (C065/91).

References

1. Springer, S., Basson, S., Spitz, J. (1992) Identification of Principal Ergonomic Requirements for Interactive Spoken Language Systems. Proc. ICSLP 92, Canada, pp. 1395-1398.
2. Siles, J.A. (1990) Telephone Speech Input/Output Applications in Spain. SPEECH TECH '90.
3. Gagnoulet, C., Damay, J. (1990) MARIEVOX: A Speech-Activated Voice Information System. CNET, rapport de stage; Lannion, 1.990, pp. 569-572.
4. Guyomard, M., Siroux, J., Cozannet, A. (1989) The Role of Dialogue in Speech Recognition. The Case of the Yellow Pages System. Proc. EUROSPEECH 89, Paris, pp. 1051-1054.
5. Young, S.J., Proctor, C. (1989) The Design and Implementation of Dialogue Control in Voice Operated Database Inquiry Systems. Computer, Speech and Language 3, 329-353.
6. Jack, M.A., Foster, J.C., Steinford, F.W. (1992) Intelligent Dialogues in Automated Telephone Services. Proc. ICSLP 92, Canada, pp. 715-718.
7. Rosenbeck, P., Baungaard, B. (1992) Experiences from a Real-World Telephone Application: teleDialogue. Proc. ICSLP 92, Canada, pp. 1585-1588.
8. Riccio, A., Carraro, F., Mumolo, E. (1989) Voice Based Remote Data Base Access. Proc. EUROSPEECH 89, Paris, pp. 561-564.
9. Mast, M., Kompe, R., Kummert, F., Niemann, H., Nöth, E. (1992) The Dialog Module of the Speech Recognition and Dialog System EVAR. Proc. ICSLP 92, Canada, pp. 1573-1576.
10. Pozas, M.J., Mateos, J.F., Siles, J.A. (1990) Audiotext with Speech Recognition and Text to Speech Conversion for the Spanish Telephone Network. Voice System Worldwide 1990.
11. Noll, A., Bergmann, H., Hamer, H.H., Paeseler, A., Tomaschewski, H. (1992) Architecture of a Configurable Application Interface for Speech Recognition Systems. Proc. ICSLP 92, Canada, pp. 1539-1542.
12. Morin, P., Junqua, J.C., Pierrel, J.M. (1992) A Flexible Multimodal Dialogue Architecture Independent of the Application. Proc. ICSLP 92, Canada, pp. 939-942.
13. Haberbeck, R. (1989) The Communication Interface - A Management System for Advanced User Interfaces. Proc. EUROSPEECH 89, Paris, pp. 573-576.
14. Lofgren, D. (1988) A specialized language for voice response applications. Proc. Speech Tech '88, vol. 2, 1, 93-94, Media Dimensions Inc., New York.
15. Boogers, W. (1989) Dialogue Construction by Compilation. Proc. EUROSPEECH 89, Paris, pp. 853-856.
16. Nielsen, P.B., Baekgaara, A. (1992) Experience with a Dialogue Description Formalism for Realistic Applications. Proc. ICSLP 92, Canada, pp. 719-722.

The Acquisition of Troubleshooting Skill

Implications for Tools for Learning

Alma Schaafstal & Jan Maarten Schraagen
TNO Institute for Human Factors
P.O. Box 23
3769 ZG SOESTERBERG
The Netherlands
E-mail: alma@izf.tno.nl

CONCEPT

1 Introduction

The task of troubleshooting or diagnosis (the terms will be defined more precisely in the next paragraph) may be described rather simply. Given that a system is not functioning properly, the troubleshooter must attempt to locate the reason for the malfunction and must then repair or replace the faulty component. Quite a lot of research has been carried out characterizing diagnostic behavior and diagnostic skill, resulting in models aimed at describing and explaining diagnostic behavior and skill. In this chapter, a theoretical framework for the interpretation of results obtained on the acquisition of troubleshooting skill wil be discussed.

Before this discussion starts it is necessary to define what is meant by diagnosis or troubleshooting, since the terms are used differently in the literature. Sometimes diagnosis is only defined as the process from identification of the symptom to the determination of the fault. In other cases, especially when one speaks about troubleshooting, the whole process of symptom identification, fault determination, and compensatory actions is taken into consideration. In this chapter, diagnosis is used in the wider sense: meaning the complete process from the identification of symptoms to the taking of appropriate corrective actions. By using this definition instead of the more restrictive one, the reader will get a more complete overview of results regarding diagnostic skill and types of knowledge used in diagnosis.

This chapter is outlined as follows. First, a theoretical framework will be discussed to enable the interpretation of results described in the literature. This will be followed by a discussion of results obtained in two domains: the paper industry and troubleshooting on board of naval ships. Finally, the implications of these findings for the development of tools for training will be discussed.

2. Diagnosis: A Theoretical Framework

Many researchers make a distinction between declarative and procedural knowledge (e.g., Anderson, 1983; 1987). Declarative knowledge may be conceived of as a collection of stored facts and is also called system or device knowledge in the domain of technical systems. Examples are knowledge about normal values of certain parameters, or knowledge about the function of the system. Procedural knowledge (knowledge about how-to-do-it) can be regarded as a collection of actions or procedures that an intelligent system can carry out. It also contains knowledge of the procedures with which one investigates a device to make diagnoses about its disfunctioning, for example the use of the oscilloscope to test certain functions of a system. Procedural knowledge is content-specific and thus only applicable in a limited domain.

In addition to the declarative-procedural distinction, a distinction can be made between domain-specific knowledge and strategic or metacognitive knowledge. This strategic knowledge (knowledge about how-to-decide-what-to-do-and-when) is applicable across specific content domains, but remains geared towards one task (e.g., diagnosis). For example, in diagnosis, irregardless of the domain, one would first *identify and interpret symptoms*, followed by *an investigation of possible reasons*, which will be *tested*,

before one will apply a certain *repair or remedy*. Metacognitive knowledge is of an even wider generality and is supposed to be both domain- and task independent. These strategies enable people to control their reasoning processes (e.g., identifying that one is on a wrong problem solving track).

At a very general level, problem-solving strategies may be identified which have to do with very general thinking and reasoning skills, such as means-ends analysis, reasoning by analogy, or working backwards (e.g., Newell & Simon, 1972). These general problem-solving strategies, or *weak methods* as they are sometimes called, are applicable across specific domains and across specific tasks.

With this decomposition, it is assumed that *procedural* and *device* knowledge are organized and deployed by goals, plans and decision rules that comprise *strategic* knowledge. Thus, strategic knowledge can be said to have a *control* function to enable dynamic, flexible reasoning. As described in Gott (1989), support for the concept of strategic control knowledge comes from a number of academic domains such as geometry (Greeno, 1978), text editing (Card, Moran, & Newell, 1983), computer programming (Anderson, Boyle, Farrell, & Reiser, 1984), simple device operation (Kieras & Bovair, 1984) and electronic troubleshooting (Gott & Pokorny, 1987). The question is, though, whether this strategic knowledge, being general in nature, can be transferred to tasks in related fields, in this manner enabling intelligent systems to generalize across domains. An example of a recent attempt to build a system that is able to generalize across domains, and in which the interaction between domain specific and domain independent strategic knowledge is explored, is FERMI, an expert system that reasons about natural science domains (Larkin, Reif, Carbonell, & Gugliotta, 1988). In FERMI, domain specific knowledge of scientific principles and strategic problem solving knowledge are encoded in separate but related semantic hierarchies. This allows the system to apply common problem solving principles of invariance and decomposition as encoded in the strategic problem solving knowledge base to a variety of problem domains such as fluid statics, DC-circuits, and centroid location. Similarly, in the knowledge based tutoring system Guidon for a medical domain (Clancey, 1987), it turned out to be very important to separate tutoring knowledge, and knowledge about general strategies, from specific domain knowledge. Processes of acquiring strategic knowledge have been addressed in analyses by Anzai and Simon (1979) on the Tower of Hanoi problem and by Anderson, Farrell, and Sauers (1984) on the acquisition of knowledge needed to learn Lisp. As stated in Greeno and Simon (1988), both studies showed that important factors in acquiring strategic knowledge are the activation of a problem goal that can be achieved by a sequence of actions and the acquisition of productions in which the action of setting the goal is associated with appropriate conditions in the problem situation. The importance of strategic knowledge has been shown by Greeno (1978). He conducted an experiment on the acquisition of high school geometry knowledge. As the resulting computational model PERDIX of problem solving behavior showed, strategic knowledge is needed for setting goals that organize problem-solving activity. One of the students in Greeno's study knew the problem-solving operators and the geometric patterns to achieve them, but was unable to solve the problem. This result can be explained by the hypothesis that the student lacked knowledge about the problem solving strategy needed in this problem. Schoenfeld (1979) showed that students who were given explicit instruction in the use of heuristic strategies in the domain of college mathematics, showed superior performance compared to students who had not received this training.

A difference exists between strategic knowledge as implemented in PERDIX or as trained in Schoenfeld's study, and strategies as used by, for example, the General Problem Solver (GPS) (Ernst & Newell, 1969) or Soar (Laird, Newell, & Rosenbloom, 1987). In GPS and Soar, only very general problem-solving strategies, such as means-ends analysis, have been implemented. These weak methods, as stated before, do not contain any domain-specificity. In PERDIX and in Schoenfeld's study the emphasis is put upon a generic strategy for a class of problems, such as diagnosis or geometry problems, which is a rather different enterprise.

Thus, if these results are put together, the following framework emerges. At the top level, strategic knowledge is employed, consisting of several goals that have to be fullfilled during task execution. Examples of these goals in the diagnostic task may be "Identify symptoms", "Determine possible causes", and "Identify repair". As the reader may have noticed, these goals may also be viewed as various subtasks of the diagnostic task. This layer of knowledge is called knowledge about the *task structure*, or the *global strategy* (Wognum, 1990). The order in which those goals are fullfilled is often flexible, and depends on the specific task-situation at hand. Each goal, however, only defines *what* intermediate conclusion has to be deduced, not *how* the intermediate conclusion has to be deduced. The

knowledge about how to perform a (sub)task is called the *local strategy*, and may consist of either procedures with a fixed order (how to test a certain part), or may consist of a more flexible sequence of steps: a strategy. The local strategy of a subtask cooperates with the *domain knowledge* necessary to achieve the goal of the task. If a person becomes very experienced with the task at hand, he may use heuristic shortcuts in his reasoning process to obtain values for the various goals determined at the task level. For example: if that symptom occurs, it is rather likely that it is caused by that fault. The heuristics are bound to the task at hand: people very experienced in one domain, which may turn them into an efficient problem-solver regarding that domain, do not usually show that same proficiency in other domains. On the other hand, if a person has no experience at all with the task at hand, he is bound to use very general reasoning methods: weak methods.

The question is what these different reasoning steps, the global strategy, the local strategies, and the underlying domain knowledge, entail in diagnosis in technical environments. The following sections will discuss data pertaining to these issues.

3. Diagnosis in Technical Environments: The Task Structure

In real-life diagnostic tasks one can distinguish between several discrete steps in the reasoning process. In a study on diagnostic reasoning of a service engineer, Jansweijer, Benjamins, and Bredeweg (1989) distinguish between two main tasks the service engineer has to perform: the diagnosis task, basically concerned with generating and testing possible hypotheses about faults in the device, and the prediction task that is concerned with deriving expected behavior of (assemblies) of components. This is a rather broad distinction though, and the question is what a finer-grained analysis of those tasks reveals.

Based on experience with the development of knowledge-based systems aimed at diagnosis, Schaafstal (1991) proposes a model consisting of a number of steps. The steps proposed show some similarity to the steps in diagnostic reasoning put forward by Wognum and Mars (1989) and Wognum (1990), but are more extended, and further elaborated. The following steps may be distinguished:

- **Identification of symptoms**

  Symptoms are the first indication that there is something wrong. This does not necessarily have to lead to an alarm situation, but it may be just a process value that is moving towards the defined limits. "Being tired" is not necessarily a symptom for a certain disease, but when you think that you are now more tired than you were before, you should start wondering about possible causes. "The machine is slightly more noisy than yesterday". In this example "being noisy" is not the symptom, but "being more noisy" may possibly be. The reader may notice that one needs expertise to interpret a certain signal as a symptom. Symptoms can be hard to describe, especially when they have a large perceptual component. In technical environments it happens quite often that operators or maintainers debate about the appearance of symptoms, or have very idiosyncratic names for symptoms, which makes uniform communication about them rather difficult.

  Depending on the domain, symptoms may be easier or harder to detect. In industrial domains it is often quite obvious that there is a problem. On the other hand, for medical domains it may be a lot harder due to vagueness of presentation of symptoms.

- **Judgment: How serious is the problem**

  Depending on a judgment about the seriousness of the problem, the whole line of reasoning and action-taking may change. If a problem is serious, it is important to take some action right away, for example to save someone's life or to prevent that any other unsafe situations will appear. It may also include a prevention of halting of the installation. When the necessary actions have been taken, the normal diagnostic routine may still take over, but that depends on the situation.

  A correct judgment in this sense is very important in many domains, especially when human safety is at risk, but also in process control situations.

- **Determination of possible faults**

  For a certain symptom or set of symptoms there is often more than one possible underlying fault. For example for quality problems in paper and board manufacturing, it is not unusual that there are more than thirty possible faults that may result in the symptom to diagnose. Evidence by Mehle (1980), and Gettys, Manning, Mehle, and Fisher (1980), cited in Morris and Rouse (1985) showed that people have difficulty in generating complete sets of hypotheses. Mechanics presented with automobile problems were generally unable to identify all of the possible causes of these problems. However, subjects were frequently overconfident about the completeness of their lists. In real-life situations, especially when "tunnel vision" appears in situations of stress, it is likely that people "forget" to take some of these possible faults into account. Especially with shift work it occurs quite often that not all operators are aware of all possible faults, due to the fact that they have not been exposed to all of them.

- **Ordering of faults according to likelihood**

  The ordering of possible faults according to likelihood is a process in which certain probabilities are "assigned" to each possible fault. In this way, an ordering of the list of possible faults appears, with the most likely candidates on top. It should be the input to the process of testing.

- **Testing**

  Testing has the function of selecting the "right" fault out of many, by ruling out as many of the other candidates as possible. Testing often takes place by collecting additional evidence, and may, especially in humans, be more like "finding extra arguments for what you already thought it would be", than "find ways to rule out this possibility" ("confirmation bias").

- **Determination of repairs or remedies**

  Repairs can take several forms. *Local* repairs apply to the exact fault at hand, and are meant to alleviate a specific problem. This is an example of *compensation for faults* (Rouse, 1982, cited in Bainbridge, 1984). *Global* repairs, on the other hand, seem to work in many situations, and can sometimes be applied without a full diagnosis of the problem. A local repair in the paper industry would be for example cleaning a dirty element. A more global repair would be slowing the speed of the paper machine down, since that seems to remedy many problems, but is in general not highly recommended. A global remedy in medicine may be "staying home for a day". This is an example of *compensation for symptoms* (Bainbridge, 1984).

- **Determination of the consequences of the application of repairs**

  In many domains it is important to realize what consequences the application of a repair might have. For example, what side-effects do certain medications have, or in a paper mill, does application of this repair imply that the installation has to be halted. The outcome of this reasoning process may have an influence on the choice of repair, if there is more than one possibility.

- **Evaluation: has the situation improved?**

  The final step in diagnosis would be an evaluation of actions undertaken in terms of improvement upon the situation. Has the problem been solved or are other actions necessary?

Based on a study of expert and novice operators in the paper industry Schaafstal (1991) concluded that the strategy used by expert and novice operators is rather different. Figure 1 gives an overview of the different global strategies an expert operator may follow.

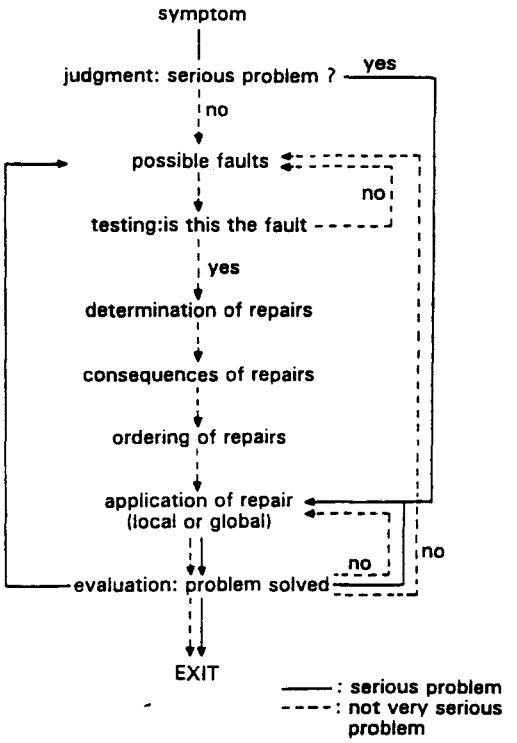


Fig. 1 Global strategies as applied by expert operators

If the problem is judged to be serious, he will immediately continue with the application of a (global) repair, followed by an evaluation whether the problem has been solved. This process may be followed by a more complete diagnosis in which the fault causing the alarm is diagnosed in order to determine the correct local repair, ensuring a solution "once and for all". If the problem is not a very serious one, the subject will consider possible faults one by one and test them, until a likely one is found. This is then followed by a determination of repairs, their consequences, an ordering of repairs (if necessary), application of repairs and an evaluation whether the problem has been solved. If not, the expert might do two things: either try another repair, or back up higher in the tree: he may realize that he has not yet spotted the actual fault, and therefore the problem has not been solved. In case no possible faults are left, or the operator cannot think of any other faults than the ones he already tested, he will be inclined to use a global repair to alleviate the problem.

The model describing the diagnostic strategy employed by relatively inexperienced operators differs from the expert model and is far more simple. Firstly, some of the categories (Judgment, Evaluation, and Consequences) are lacking altogether. Also novices jump much more quickly to repairs, without realizing whether a certain repair actually is right and optimal for a certain situation. Figure 2 gives an overview of a model for the diagnostic strategy employed by inexperienced operators.

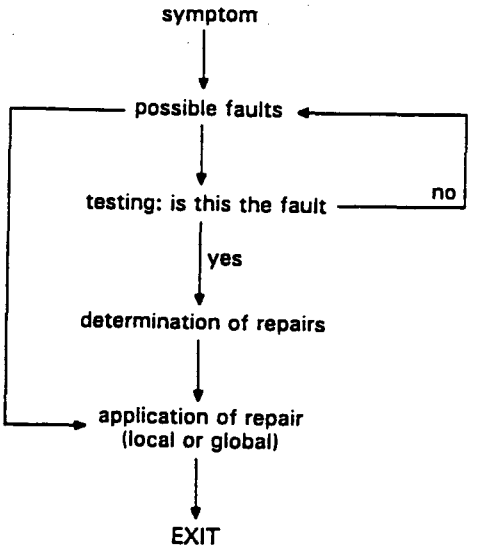


Fig. 2 Global strategies as applied by inexperienced operators

In an electronics troubleshooting domain, Schraagen en Schaafstal (1991) found similar problems for novice maintenance technicians. One of the conclusions of that research is that novices need to develop robust, flexible troubleshooting strategies that are based on a functional understanding of the system.

Thus, in conclusion, the task structure for diagnosis in technical environments may be different for experts and novices in a certain domain. If one wants to use the task structure as in interpretation scheme for the development of tools for training, the task structure used by expert subjects may serve as the normative task structure (the goal to work towards), and can be used as such.

4. Diagnosis in Technical Environments: Local Strategies

Much of the work that may be interpreted as investigating local strategies in diagnosis has to do with local strategies that are used to determine the fault given the set of symptoms. In terms of the task structure, this fault finding process includes the process of testing, and an ordering of the faults according to likelihood. As described by Jansweijer (1989), sometimes symptoms are strong indicators for certain faults and relate the symptom more or less directly to the remedy. Often these relations are indirect. In that case, data obtained from observing the installation will have to be abstracted before they can be associated with possible causes for this malfunctioning, or a possible solution has to be refined to an executable remedy. This type of reasoning is known as *heuristic classification*. A slightly different association is the one where a symptom can be translated into disturbed functions. For example, a puddle of water underneath the washing machine probably leads to the conclusion that the water-contain function of the washing machine is disturbed. It is best to focus on that function and to exclude other functions such as the timer-control or motor-function. A third type of knowledge that is used to interpret symptoms is knowledge about the expected appearance of the "inside" of the device. The physical appearance of parts and components is compared to an expected outlook. A loose wire has to be connected again. Otherwise symptoms such as, for instance, a hot or burnt component, smoke, sparks or a strange sound give an immediate focus for diagnosis. As for all these three types of shallow knowledge used to interpret symptoms into a certain direction, they may be misleading, in a sense that a component may look all right, but in fact is not (such as may happen with plugs).

Quite relevant with respect to search strategies from symptom to underlying fault is the work by Rasmussen and coworkers and Rouse and his colleagues. This work will now be discussed at some length, before continuing with what is known about other search strategies, such as the ones used for testing.

Rasmussen

Rasmussen's basic study of diagnostic strategies was carried out in an electronic instrument repair shop under real-life conditions (Rasmussen & Jensen, 1974), using interviews and verbal protocols as research tools. The results from this study have later been generalized to diagnosis in computer systems and process plants, together with an analysis of error reports from power plants. The most important findings of the study on technicians in the electronic repair shop were the following, as described in Rasmussen (1986):

- The trained technician, contrary to the designer, used many observations in a sequence of simple decisions. He uses a general search procedure that is not dependent on the actual system or specific fault. He treats the observations individually in a stream of good/bad judgments that is informationally uneconomic, since no previously experienced faults and disturbances are taken into account and only good/bad judgments are made. However, these observations are made very fast, and thus, it may pay off to make a few more observations;
- The technician defines his task primarily as a search to find where the faulty component is located in the system. He is not concerned with explanations why the system has the observed faulty response or understanding the actual functioning of the failed system;
- Search procedures are organized as a search through a system that is viewed as a hierarchy of units and subunits;
- The general structure of the search can be broken down into a sequence of three different search routines, which are used to identify the appropriate subsystem, stage, or component. These three search strategies employed by the technicians are *functional search*, *topographic*

search, and *symptomatic search*. In functional search, the topographic reference is obtained from the normal functional relationship between a feature in the system response and a specific part of the system. This search is a special type of the topographic search as discussed below. The information pattern is scanned and familiar features are judged individually in a stream of good/bad judgments. If a response feature is judged faulty, attention is typically turned immediately toward the subsystem related to that function. In topographic search, reference to the location of the fault is obtained from the topographic location of a measuring point. In symptomatic search, a set of observations representing the abnormal state of the system - a set of symptoms - is used as a search template to find a matching set in a library of symptoms related to different abnormal system conditions. Symptomatic search consists of a form of pattern-matching between the symptoms and the result in terms of a label, which may be a cause, effect, location, or an appropriate control action. Symptomatic search is advantageous from the point of view of information economy, and a precise identification can frequently be obtained in a one-shot decision. Emphasis is put upon immediate knowledge, based on stored information. A serious limitation is that a reference pattern of the actual abnormal state of operation must be available, and multiple faults and disturbances may be difficult to take into account. Reference sets must be prepared by analysis or recorded from prior occurrences (role of experience!). Symptomatic search may be considered a form of heuristic reasoning .

A very interesting feature of the study on electronic troubleshooters, as described in Rasmussen (1986) is the fact that technicians showed a pronounced ability to use general search routines that are not closely related to the specific instrument. Scanning a high number of observations by simple procedures is preferred to the preparation of specific procedures worked out by studying or memorizing the internal functioning of the system. Thus, these data indicate that people may have domain-independent diagnostic strategies that possibly can be used in a wide variety of domains.

Interesting in Rasmussen's approach is that the important distinction that he makes is between context-specific strategies (symptomatic search) vs. context-free strategies (topographic search). Thus, Rasmussen's distinction is based on the level of generality of search strategies. Context-free search strategies will be widely applicable, while symptomatic search strategies are tied to one specific context for their application. In AI-approaches it is more common to make a distinction between context-specific rules-of-thumb (heuristics) and, when heuristics are unavailable, the use of model-based reasoning, which is still domain-dependent reasoning, coupled with rather weak reasoning methods. Examples of these weak reasoning methods are a backward strategy for elimination of suspects, a forward strategy for elimination of suspects, and an elimination strategy to split the set of suspects into two halves (Jansweijer, 1989). Thus, AI-approaches are more geared towards a distinction between knowledge and search, while Rasmussen is more focussed towards differences in generality of search strategies.

Rouse

The discrepancy between the supposed human abilities to react appropriately and flexibly in failure situations and the occurrence of "human error" by which failure situations may be aggravated has been one of the reasons for Rouse and coworkers to start a research program on human problem solving performance in fault diagnosis tasks. A series of experiments was carried out, and a number of mathematical models for human problem solving was developed, which are described in Rouse (1982), and Rouse and Hunt (1984). What these researchers were particularly interested in is the theoretical question whether diagnostic skills are context-free or context-specific, which has, among others, huge implications for training issues. Thus, their work shows similarities in this respect to Rasmussen's work.

The experimental work was done using four different fault diagnosis tasks. Two of them, TASK1 and TASK2, were context-free diagnosis tasks (computer simulations of graphically displayed network representations) in which no association exists between the tasks and a particular system or piece of equipment. These context-free network representation tasks enable diagnosis based on the structure of the network, and may therefore be considered the psychological pendant of model-based reasoning based on descriptions of the structure of a system. Since in these tasks used by Rouse there is no connection with a particular piece of equipment or system, it is impossible to develop context-dependent skills. However, one would like to know whether context-free skills can be used in context-specific tasks, and

the question is whether subjects can be trained to have general skills that are transferable to context-specific situations. Therefore a third fault diagnosis task was devised, FAULT, in which hardcopy schematics of various systems, such as automobile, aircraft, and marine systems can be employed. In this way, context-specificity is ensured. Finally, a number of experiments was carried out using real equipment in which subjects were required to diagnose failures in four and six cylinder engines. Apart from the various tasks, different forms of aiding were constructed. The results of experiments carried out with these tasks showed that in general, context-free skills learned with computer aiding in one task can be successfully transferred to another context-free or context-specific task. Positive transfer of training can be explained as a reordering of priorities within a set of basic problem solving rules: people know which rules to use in a certain situation. This reordering appears to enable trainees to utilize the structure of the problem to a greater degree and thereby make more efficient tests in the sense of achieving greater reductions of uncertainty per unit cost. Performance on context-free tasks is highly correlated with performance on familiar real equipment tasks, which may imply that the use of a good strategy is at least as important as the availability of domain knowledge (knowledge about the equipment).

Based on the results of the experiments, Rouse and coworkers developed a number of models for human problem solving. The first model was the fuzzy set model, based on the idea that people may not always be able to determine with absolute certainty whether a certain component could cause a particular set of symptoms. The second model, the rule-based model, was based on the assumption that fault diagnosis involves the use of a set of heuristics from which a person selects, using some type of priority structure. However, the success of this rule-based model was limited to context-specific situations, in which subjects shift from topographic to symptomatic search, to use Rasmussen's terminology. These two models were therefore integrated into the fuzzy rule-based model, which is a formalization of the dichotomy between symptomatic and topographic search. This fuzzy set rule-based model first attempts to find familiar patterns among the symptoms. If a match is found, symptomatic rules (S-rules) are used to map directly from symptoms to hypothesized failure. If there are no such familiar patterns, topographic rules (T-rules) are used to search the structure of the system. Thus, when heuristics are available, they are used. If not, subjects have to rely on a general search strategy such as topographic search, which is context-free.

Some of the different search strategies that have been described in the literature may be considered more powerful strategies than others. Obviously, heuristics (or symptomatic search), connecting symptoms to underlying faults, belong to the most powerful strategies an operator may have, although they are only applicable in a narrow domain, and have no wider generality beyond this domain. Therefore, they are likely to fail in any new situation. Less powerful, but still leading to conclusions rather efficiently, are search strategies such as topographic search, geared towards diagnosis in technical domains, but more widely applicable than heuristic search. At the next level of generality are search strategies such as split-half approaches, in which the goal is to minimize the number of tests to localize the source of the failure. Experiments by Goldbeck, Bernstein, Hillix, and Marx (1957) showed that training subjects to make use of a split-half approach only enhanced performance in simple systems. This is understandable since a split-half approach, due to its generality, lacks the power to handle complex systems efficiently, but has to be supplemented with knowledge about how to divide a system in various subsystems. In simple systems, though, one may not need such powerful strategies to be able to come up with solutions efficiently. At the lowest level of specificity are so-called weak problem solving methods, such as means-ends analysis and generate and test, as described by Newell and Simon (1972). These search mechanisms are asserted to be a central component of general problem-solving skill and are very general in scope, thus trading power for generality.

The search strategies used by technicians may be different in different situations, and may also depend on the level of expertise. Since heuristics are mostly developed through practical experience and are tied to specific situations, they may become increasingly available with increasing levels of expertise. As demonstrated by Rasmussen (1976), topographic search strategies, although not informationally economic, may be preferred by technicians in domains such as electronic troubleshooting.

Although most of the research on search strategies focusses on fault finding, some research, such as by Goldbeck et al. (1957) discussed above, has been done investigating the strategies people use to optimize the number of tests needed to localize the source of the failure. Interesting in this respect are experiments carried out by Morrison and Duncan (1988), in which people were presented with novel failures in a context-free network task. According to these researchers, this task is comparable to

electronic circuitry troubleshooting or troubleshooting with respect to powerlines in a telecommunications system. They found that in this task it was more efficient in terms of number of tests if subjects spent more time on utilizing overview information on the system, instead of simply testing units sequentially from right to left. However, it should be noted that the former strategy puts a much higher demand on working memory resources, which are only limited available, and secondly, that the less efficient strategy did not necessarily lead to a lower diagnostic performance.

5. Diagnosis in Technical Environments: Domain Knowledge

Now that we have adressed what has been found about the task structure in diagnosis and the corresponding local strategies, the question remains what kind of domain knowledge plays a role in diagnosis in technical environments. Early work on expert systems seemed to imply that there was only one underlying model of domain knowledge. Work along this line gave rise to the *model-based approach*, which postulates that expert system building should start with an encoding of the first principles of a domain, for example, qualitative or quantitative models of the behavior of the device to be diagnosed (Davis, 1984; DeKleer, 1984). In most of the cases, this work concentrates on models about the structure and behavior of the device. However, as described in Steels (1990), it is possible to think of a variety of models, each focussing on different aspects of the problem domain. For diagnosis, for example, one could think of a structural model describing part-whole relationships between components and subsystems, a causal model representing the cause-effect relationships between properties of components, a functional or behavioral model representing how the function of the whole follows from the function of the parts, a fault model representing possible faults and components for each function that might be responsible for the fault, and an associational model relating observed properties with states of the system. Which of these models would be the one to use? Clearly, all these models are useful. Simmons (1988), cited in Steels (1990), for example, describes a system that translates causal models into associational models and shows how they have complementary utility in problem solving. Thus, the question is not what model to use exclusively, but what type of model or type of domain knowledge is appropriate in certain stages of the diagnostic reasoning process.

Most of the research on types of domain knowledge used concentrates on the process of fault finding. Jansweijer et al. (1989), for example, found that the type of model that is of primary use to the service engineer of the UV-recorder is a model of the *function* of the device on different levels of abstraction. The service engineer knows the set of subfunctions that realizes a higher level function. The subfunctions can either be a set of independent subfunctions or a series of subfunctions that enable each other. Each subfunction can itself be described in the same way until, at the lowest level, a function is realized by a component that cannot be decomposed further, or of which the decomposition is of no interest since it is not repairable, but has to be replaced entirely.

In this way, a multi-fold representation of the device on different levels of abstraction exists. On the highest level functions are described with global parameters covering functions as expected by a naive user. At the lower level, the functions are described with parameters covering more detailed and inner functions of the device. A high level function of a washing machine for example is "the cleaning of dirty wash", with a parameter of the "cleanliness" of the linen. A lower level function of a washing machine for example would be heating the water.

The knowledge about the domain that the service engineer exploits is sometimes of a generic type. On a high level, knowledge about the function is device independent and independent of the physical realization. A service engineer that knows to locate and replace, for instance, a defective water-pump in a washing machine, knows to do the same in a washing machine of a different brand. Moreover, he is presumably able to locate and repair the same component in a machine with a somewhat different high-level function, such as a dish-washer. Having knowledge about high-level generic functions is an efficient way for representing knowledge about systems since it can be used in other, different domains.

To summarize: the multi-fold representation of the device, on different levels of abstraction has a hierarchical structure. Higher functions are realized by the subfunctions it subsumes. High levels describe functions independent of the physical realization. On the lowest level, the model corresponds to a static description of all the components and the relations between these components. On this level the service engineer usually has to depend on service manuals and schemata. But this level of representation is usually too detailed for the technician to be of any use and seldom needed as well. In standard repair

practice, complete assemblies of components will be replaced rather than the individual components.

If we take a more detailed look at the types of knowledge people have about devices it is clear that people know a lot more about the device than just the functioning and the structure of a device. For example, Kieras (1982, 1987) showed that people have at least the types of knowledge available as described in table 1.

Table 1 *Types of knowledge people have about devices (adapted from Kieras, 1982).*

The label or name of the device
The functions or purpose (what goals can be accomplished)
The controls and indicators
The inputs, outputs, and connections
Power source and requirements
External layout and appearance
Internal layout and appearance
External behavior (input-output function)
How to operate the device to accomplish goals
Procedures for troubleshooting and maintenance
Internal structure and mechanisms (how it works)

Table 1 suggests that most knowledge about devices is related to *using* the device, as opposed to *how-it-works knowledge* about the internal structure and operation of the device. For example, Kieras (1982) observed that when asked to freely describe a device, experts would provide considerable detail on the procedure for using a piece of equipment, but often they did not consider it necessary to provide any details about how the device worked, although it was very clear that they would be able to do so when asked. The question is, though, whether the types of knowledge people use and report depends on their job-contents (an operator vs. a technician, or a technician vs. a manager), and there is suggestive evidence in the literature that this is in fact the case (Cuney, 1979; Rasmussen & Lind, 1981; Schaafstal, 1989).

If we take a closer look at the relationship between various types of underlying domain knowledge and stages in the diagnostic process as described in section 2, the following relationship is suggested by Schaafstal (1991), taken from an industrial domain (the paper and pulp industry):

Table 2 *Relation between phases of the model and used system knowledge.*
1 = Symptoms, 2 = Judgment, 3 = Possible faults, 4 = Ordering of faults, 5 = Testing, 6 = Determination of repairs, 7 = Consequences of application of repairs, 8 = Evaluation.

	1	2	3	4	5	6	7	8
Proces flow		*	*		*		*	
Top. location						*		
Controls	*		*		*	*	*	
Function comp.	*	*	*			*	*	
Paper making	*	*	*	*		*	*	*
Normal values	*		*		*			*
Process dynamics	*		*		*			*
Functioning comp.			*			*		

The second domain of study, electronics troubleshooting in a radar system also showed that people use a wide variety of system knowledge, in the same vein as the operators in the paper industry (Schraagen & Schaafstal, 1991). However, much more research in different domains is still needed to establish more firmly this kind of interaction between types of domain knowledge and stages in the diagnostic reasoning

process.

This section may be concluded by the following remarks. Much of the earlier work on model-based reasoning focussed on models of the structure and behavior of the device. Although this work has lead to very interesting insights, in the long run this seems to be a too restrictive viewpoint, as pointed out by Steels (1990). Thus, a more flexible view towards the use of domain knowledge is needed, certainly if we take into account the flexiblity with which people use different types of knowledge, at different levels of abstraction and presumably also at different stages of the diagnostic reasoning process.

6. Implications for the training of troubleshooting skill

One of the most striking findings in the literature, as summarized in Morris and Rouse (1985) is that instruction in theoretical principles is not an effective way to produce good troubleshooters. It is interesting to note that these results are quite consistent with reports from other domains such as process control (Brigham & Laios, 1975; Crossman & Cooke, 1974; Kragt & Landeweerd, 1974; Morris & Rouse, 1985; Mayer & Greeno, 1972; Mayer, Stiehl, & Greeno, 1975), in which explicit training in theories, fundamentals, or principles failed to enhance performance, and sometimes actually degraded performance.

However, when theoretical instruction is combined with training people in how to use that knowledge, performance usually gets better. For example Miller (1975) described an experimental training course for radar mechanics, in which an effort was made to relate instruction in system functioning to actions performed during troubleshooting. For example, system behavior was presented in terms of causal sequences rather than a more traditional presentation of schematics. Whenever possible, attempts were made to relate schematics to the actual equipment. A control group received instruction in the theory upon which the system was based and a left-to-right presentation of schematics, with no extraordinary attempt to relate the information to the actual equipment. Both groups had limited access to the actual radar for troubleshooting practice. The experimental group turned out to be faster in performing checks and adjustments, was more often successful in troubleshooting, made fewer errors in general, and scored better on quizzes.

It should be noted, though, that in the studies in which positive effects were found, the guidance involved was rather explicit: students were told to generate hypotheses, chunk information, and analyze symptoms in a prescribed way. This is a far more active approach than just providing an opportunity to use system knowledge, and should not be interpreted as evidence that the latter approach will produce better troubleshooters.

Thus, one of the elements that helps in training people to become better troubleshooters is an explicit guidance in the use of previously acquired theoretical knowledge. Another important factor in training of troubleshooting is the opportunity for practice (e.g., Johnson & Rouse, 1982a, 1982b), which is also strongly advocated by many researchers in intelligent tutoring systems (e.g., Lesgold, 1992).

Not much is known about training in local strategies. A number of studies showed that supplying people with adequate procedures can have a positive effect on their troubleshooting performance (Potter & Thomas, 1976; Smillie & Porta, 1981; Elliott, 1966). There is also limited evidence available that providing troubleshooters with good examples can have a beneficial effect on their performance (e.g., Johnson & Rouse, 1982a, 1982b). However, learning from examples may be confined to people with high ability, and appears to be dependent on explicit instruction to learn from the examples as well.

Training with respect to task structures is still a rather neglected area
This may partly be due to the misconception that training in domain, or system, knowledge will automatically result in good troubleshooting performance, since the two are closely linked. It may also stem from the idea that good troubleshooting skills will automatically evolve with experience on-the-job, and therefore explicit training will not help all that much: experience will do the work. In itself: this idea is valid: the expert troubleshooters in our studies (Royal Dutch Navy and paper industry) became fine diagnosticians without explicit strategy training. However, the question is whether training in diagnosis as a whole can be speeded up and be made more efficient if strategy training is taken into account. A final reason for the absence of strategy training is the fact that good strategy training is difficult to accomplish and involves quite some analysis before good strategies have been identified and made sufficiently explicit for the incorporation in regular training courses.

7 Implications for the Development of Tools for Learning

The training of strategies requires a learning environment with sufficient possibilities for practice, and guidance in using a good strategy. This is not always easy to accomplish in real-life settings, since due to risks involed in those situations it is not always possible to freely experiment with installations or devices. A second problem involved is that in a real-life setting one has no control over problems that occur, and therefore it is hard to establish a training program solely on the basis of what happens in practice. Therefore, ideally one would need training tools and training environments that enable a trainee to systematically work through series of problems that have been controlled with respect to the current focus of training and in which appropriate, individualized, feedback can be given as well. In this way, it becomes possible to devise flexible learning tools, that enable individual trainees to follow their own learning trajectory. Only now, these environments become available, often as a result of the application of AI-oriented research. Third, the relationship between the training of domain knowledge and the training of strategies how to apply that knowledge is not very well understood. Recent efforts towards the development of coached practice environments for the training of troubleshooting, such as demonstrated by the Sherlock project (Lesgold and coworkers, 1992), are rather promising though in terms of their training results, and suggest that strategy training in relation to the acquisition of the relevant domain knowledge is possible and worth the effort.

Thus, to conclude with: strategy training, although important, is a neglected area in the training of troubleshooting skill, which certainly deserves more attention. This attention is now gradually growing, partly based on technological improvements. However, before strategy training, as part of the training of troubleshooting skill as a whole can be accomplished, there is a serious need for good methods for accomplishing cognitive task analyses, such that the strategies used and knowledge involved is made explicit. The method summarized in this chapter, but more fully discussed in Schaafstal and Schraagen (1992) is meant to be a contribution to this issue.

References

Anderson, J.R. (1983). *The architecture of cognition.* Cambridge, MA: Harvard University Press.

Anderson, J.R. (1987). Skill acquisition: Compilation of weak-method problem solutions. *Psychological Review, 94,* 192-210.

Anderson, J.R., Boyle, C.F., Farrell, R.G., & Reiser, B.J. (1984). Cognitive principles in the design of computer tutors. In *Proceedings of the Sixth Cognitive Science Society Conference.* Boulder, CO.

Anderson, J.R., Farrell, R.G., & Sauers, R. (1984). Learning to program in LISP. *Cognitive Science, 8,* 87-129.

Anderson, J.R., & Reiser, B.J. (1985). The LISP tutor. *Byte, 3,* 159-175.

Anzai, Y., & Simon, H.A. (1979). The theory of learning by doing. *Psychological Review, 86,* 124-140.

Bainbridge, L. (1984). *Diagnostic skill in process operation.* Revised version of invited review paper presented at the international conference on occupational ergonomics.

Bonar, J.G. (1985). *Mental models of programming loops.* Technical report, Pittsburgh, PA: University of Pittsburgh, Learning Research and Development Center.

Brigham, F., & Laios, L. (1975). Operator performance in the control of a laboratory process plant. *Ergonomics, 18,* 53-66.

Card, S., Moran, T.P., & Newell, A. (1983). *The psychology of human-computer interaction.* Hillsdale, NJ: Lawrence Erlbaum.

Clancey, W.J. (1987). *Knowledge-based tutoring. The GUIDON program.* Cambridge, MA: MIT Press.

Crossman, E.R.F.W., & Cooke, J.E. (1974). Manual control of slow-response systems. In E. Edwards & F.P. Lees (Eds.), *The human operator in process control.* Londen: Taylor & Francis.

Cuney, X. (1979). Different levels of analysing process control tasks. *Ergonomics, 22,* 415-425.

Davis, R. (1984). Diagnostic reasoning based on structure and behavior. *Artificial Intelligence, 24,* 97-130.

DeKleer, J. (1984). How circuits work. *Artificial Intelligence, 24,* 205-281.

Elliott, T.K. (1966). A comparison of three methods for presenting troubleshooting information. Tech. Report AMRL-TR-66-191. Valencia, PA: Applied Science Associates.

Ernst, G.W., & Newell, A. (1969). *GPS: A case study in generality and problem solving.* New York:

Academic Press.
Goldbeck, R.A., Bernstein, B.B., Hillix, W.A., & Marx, M.H. (1957). Application of the half-split technique to problem-solving tasks. *Journal of Experimental Psychology, 53*, 330-338.
Gott, S.P. (1989). Apprenticeship instruction for real-world tasks: The coordination of procedures, mental models, and strategies. In E.Z. Rothkopf (ed.), *Review of Research in education, 15*. Washington, DC: AM. Educ. Res. Assoc.
Gott, S.P., & Pokorny, R. (1987). The training of experts for high-tech work environments. In *Proceedings of the Ninth Interservice/Industry Training Systems Conference.* Washington, DC.
Greeno, J.G. (1978). A study of problem solving. In R. Glaser (Ed.), *Advances in instructional psychology: Vol. 1.* Hillsdale, NJ: Lawrence Erlbaum.
Greeno, J.G., & Simon, H.A. (1988). Problem solving and reasoning. In R.C. Atkinson, R.J. Herrnstein, *cognition (2nd edition)*. New York: John Wiley & Sons.
Jansweijer, W.N.H., Benjamins, R., & Bredeweg, B. (1989). Diagnostic reasoning of the service engineer. Paper submitted to Conference on Second Generation Expert Systems, Avignon, 1989.
Johnson, W.B., & Rouse, W.B. (1982a). Analysis and classification of human errors in troubleshooting live aircraft powerplants. *IEEE Transactions on Systems, Man, and Cybernetics, SMC-12*, 389-393.
Johnson, W.B., & Rouse, W.B. (1982b). Training maintenance technicians for troubleshooting: Two experiments with computer simulations. *Human Factors, 24*, 271-276.
Kieras, D.E. (1982). *What people know about electronic devices: A descriptive study.* Technical report No. 12. UARZ/DP/TR-82/ONR-12. University of Arizona, Department of Psychology.
Kieras, D.E. (1987). *What mental model should be taught: Choosing instructional content for complex engineered systems.* University of Michigan, Technical Report No. 24.
Kieras, D.E., & Bovair, S. (1984). The role of a mental model in learning to operate a device. *Cognitive Science, 8*, 255-273.
Kragt, H., & Landeweerd, J.A. (1974). Mental skills in process control. In E. Edwards & F.P. Lees (Eds.), *The human operator in process control.* Londen: Taylor & Francis.
Laird, J.E., Newell, A., & Rosenbloom, P.S. (1987). SOAR: An architecture for general intelligence. *Artificial Intelligence, 33*, 1-64.
Larkin, J.H., Reif, F., Carbonell, J., & Gugliotta, A. (1988). FERMI: A flexible expert reasoner with multi-domain inferencing. *Cognitive Science, 12*, 101-138.
Lesgold, A. (1992). Going from Intelligent Tutors to Tools for Learning. In C. Frasson, G. Gauthier, & G.I. McCalla (red.), *Intelligent Tutoring Systems. Second International Conference ITS'92.* Berlin: Springer-Verlag.
Lesgold, A., Lajoie, S., Bunzo, M., & Eggan, G. (1992). SHERLOCK: A coached practice environment for an electronics troubleshooting job. In J.H. Larkin & R.W. Chabay (red.), *Computer-assisted instruction and intelligent tutoring systems. Shared goals and complementary approaches.* Hillsdale, NJ: Lawrence Erlbaum.
Mayer, R.E., & Greeno, J.G. (1972). Structural differences between learning outcomes produced by different instructional methods. *Journal of Educational Psychology, 63*, 165-173.
Mayer, R.E., Stiehl, C., & Greeno, J.G. (1975). Acquisition of understanding and skill in relation to subjects' preparation and meaningfulness of instruction. *Journal of Educational Psychology, 67*, 331-350.
Miller, E.E. (1975). Instructional strategies using low-cost simulation for electronic maintenance. Tech.Rep. HumRRO-FR-WD(TX)-75-20. Alexandria, VA: Human Resources Research Organization.
Morris, N.M., & Rouse, W.B. (1985). Review and evaluation of emprirical research in troubleshooting. *Human Factors, 27*, 503-530.
Morrison, D.L., & Duncan, K.D. (1988). Strategies and tactics in fault diagnosis. *Ergonomics, 31*, 761-784.
Newell, A., & Simon, H.A. (1972). *Human problem solving.* Englewood Cliffs, NJ: Prentice-Hall.
Potter, N.R., & Thomas, D.L. (1976). *Evaluation of three types of technical data for troubleshooting.* Tech Rep. 76-74(3). Brooks, AFB, TX: Air Force Human Resources Laboratory.
Rasmussen, J. (1986). *Information processing and human-machine interaction. An approach to cognitive engineering.* Amsterdam: Elsevier Science Publishers.
Rasmussen, J., & Jensen, A. (1974). Mental procedures in real-life tasks: A case study of electronic troubleshooting. *Ergonomics, 17*, 293-307.

Rasmussen, J., & Lind, M. (1981). Coping with complexity. In *European Annual Conference on Human Decision Making and Manual Control*. Delft: University of Technology.

Rouse, W.B. (1982). A mixed fidelity approach to technical training. *Journal of Educational Technology Systems, 11*, 103-115.

Rouse, W.B., & Hunt, R.M. (1984). Human problem solving in fault diagnosis tasks. In W.B. Rouse (Ed-d.), *Advances in man-machine systems research*. London: JAI Press.

Schaafstal, A.M. (1989). Kijk op procesinstallaties en kennis van storingen. In *Proceedings AI-Toepassingen '89*.

Schaafstal, A.M. (1991). *Diagnostic skill in process operation. A comparison between experts and novices*. Phd-thesis, Rijksuniversiteit Groningen.

Schaafstal, A.M. & Schraagen, J.M.C. (1992). A method for cognitive task analysis. IZF rapport 1992 B5.

Schoenfeld, A.H. (1979). Explicit heuristic training as a variable in problem-solving performance. *Journal of Research in Mathematics Education, 10*, 173-187.

Schraagen, J.M.C., & Schaafstal, A.M. (1991). Diagnose van storingen in een sewaco-deelsysteem: de LW08 radar. IZF rapport 1991 A36.

Simmons, R. (1988). *Generate, test and debug: a paradigm for solving interpretation and planning problems*. PhD-thesis, AI lab, Massachusetts Institute of Technology.

Smillie, R.J., & Porta, M.M. (1981). Comparison of state tables and ordnance publication for troubleshooting digital equipment. Tech. Report AFHRL-TR-74-57. Wright-Patterson Air Force Base, OH: Air Force Human Resources Laboratory.

Steels, L. (1990). Components of expertise. *AI Magazine, summer 1990*, 28-49.

Wognum, P.M. (1990). *Explanation of automated reasoning: How and why?* PhD-thesis University of Twente.

Wognum, P.M., & Mars, N.J.I. (1989). Het belang van uitleg van redeneringen. In L. Siklossy & R.R. Bakker (Eds.), *Proceedings NAIC '89*. Schoonhoven: Academic Service.

A Visual Knowledge Elicitation Language and Methodology for Acquiring Non Verbal Expertises

Benjamin Singer[1] and Jean-Luc Soubie[1 2]

[1] ARAMIIHS
CNRS UMR 115
31, rue des Cosmonautes
31077 Toulouse Cedex

[2] IRIT
CNRS URA 1399
118, route de Narbonne
31062 Toulouse Cedex

Abstract. In this paper we propose a Visual Knowledge Elicitation language and methodology for picture understanding. Indeed we have to take into account visual expertises for interpreting static pictures or dynamic image sequences. The expertise domain here considered is that of team games, with an application to the rugby area. The expert chosen (Pierre Villepreux and René Deleplace) propose respectively verbal and non verbal models for modelling the deep and surface knowledge. Such models are as well theoretical as practical. As a matter of fact for acquiring such a kind of knowledge sources, we developped a dedicated visual acquisition software tool, driven by the Visual Knowledge Elicitation language. Such a software is considered as an extension of the MACAO acquisition method and software. More precisely, the language helps the expert to explicit the objects and sets of objects he manipulates visually during his reasoning, the analytical geometrical models he uses, and the displacements and trajectories he draws. At the same time this language helps the knowledge engineer for the representation and structuration step.
By using this Visual Knowledge Acquisition software and by extending the MACAO method and software, we have produced the complete Visual Conceptual Model of the non verbal expertise in the considered domain. Our aim is indeed to design an educational system to help in a tactical way both the players themselves for the next games and the members of technical staff (coaches, technical director), thanks to a Knowledge-Based System running on a micro-computer.

1 Introduction

In this paper we propose a specific Visual Knowledge Elicitation (VKE) language for images or image sequences understanding. This domain independent language is included in a knowledge acquisition software tool implemented on a colour Sparc station.

The first part of the paper details our scientific problematic in distinguishing and comparing verbal and visual approaches of Knowledge Acquisition (KA). Non verbal approaches are illustrated by means of the associated more striking methodologies, which in general both include KA methods and software tools. The knowledge partition in deep and surface levels, according to Steels [16], is then described to justify our restriction to acquire only surface knowledge.

The second part of the paper relates our contribution by a specific acquisition technique, based on a VKE language, applied to interpreting symbolic image sequences of team games, with the example of rugby.

The third part deals with the representation and structuration stages, for the verbal and visual knowledge, following the MACAO methodology and using the MACAO software tool [2].

In conclusion, we point out the originality of our KA approach, and the generality of the results achieved for the growing number of similar experienced tasks, which nowadays include a major picture understanding activity.

2 Research framework : non verbal KA methodologies

Our approach of Knowledge Acquisition

The crucial problem for designing Knowledge-Based Systems (KBSs) consists in constructing its Knowledge Base (KB) [5]. In the sense of expertise transfer, the knowledge engineer work can be seen as extracting and implementing the know-how of an expert in a given domain in the framework of a specialized problem solving system [4].

A lot of experiences in constructing operational KBSs have been carried out, but after the integration phase, we could point out a lack of interest or even a non use. This fact comes from to the non satisfactory construction of the KB which is often incomplete

or misfunctioning. So that a lot of methodologies have been proposed to help the knowledge engineer in giving him a better access to any expertise domain. For instance, let us mention the KADS, KOD, 3DKAT, COPILOTE, COMMET or MACAO methods.

The main goal of KA is to propose a set of appropriated techniques and tools to help the knowledge engineer during his expertise transfer task, via in practice interactive editors. KA can not be reduced to a simple retranscription process, that of the experts specific knowledge. Above all KA is to be seen as a given domain formalization and a modelling activity [11]. The conceptual model constructed by the knowledge engineer has to translate the expert reasoning and should be easily fit for use in order to be explained. KA is seen here both as a capitalization and as an explaination process.

2.1 From verbal to non verbal methodologies : the example of KADS

KADS (Knowledge Acquisition and Design Support) is typical of top-down methodologies i.e. driven by an abstract scheme of the task model realised by the expert.

Nowadays the KADS methodology is indeed considered as a KA European standard, because of its intrinsic characteristics and its development and realisation cycle in the associated Esprit projects (KADS First and Second Version). The software engineering approach here followed has common points with knowledge engineering [18].

Moreover in KADS-TOOL [1] and OPEN-KADS (of Bull), which are toolboxes for implementing the KADS methodology, visual knowledge is manipulated by following an hypertext approach. Non verbal informations are indeed combined within the textual input data coming from the elicitation phase. They are stored in the corresponding files with the natural language description of the expertise. But they are not really processed. So the non verbal knowledge is only static in this case, and the input files are mixed and composed of textual and visual informations.

Furthermore thanks to the hypertext approach, if the user has selected a picture, the associated concept is automatically given in natural language.

So the major advantage of this methodology is that it is generic and has generated interesting workbenches which include nowadays non verbal knowledge.

2.2 Limits and inadequacy of verbal acquisition methodologies

The today's recognized and used KA methodologies, which follow top-down or bottom-up approaches, are in fact only verbal i.e. are based either on retranscripting the experts verbalisations (coming from interviews centered or not) or on already existing written documents for capitalising the expertise.

But a lot of new needs are coming up nowadays, particularly these related to the growing number of experienced technical tasks integrating an image or image sequence interpretation activity, such as medical or satellite imagery for instance. So designers are lead to propose new non verbal KA methods and realise the associated software tools. Thus we could use the term of Visual Knowledge Acquisition (VKA).

The specific methodology here proposed has been applied to the field of team games study, and particularly to understanding rugby image sequences issued of real matches. Indeed to any game sequence is associated a video colour image sequence following an adapted recording technique : a unique fixed camera with an invariant lens, placed on a crane in order to record continually all the actors (ball, single players, groups of players, and so forth).

For the domain taken into account, our study is restricted to acquire the surface knowledge coming from the Pierre Villepreux's model [17], because the deep knowledge has been already correctly formalized within the logical Deleplace's model, also called in action Tactical Choice Systematics [7]. At any time t of a game sequence, this model gives indeed all the possible options at $(t + \Delta t)$ in an n-ary decision set formalism. Such pseudo-algorithms are similar to the n-ary tree data structures.

3. Our non verbal approach based on a Visual Knowledge Elicitation language

Context of our works

Our study context is that of image sequence understanding in the domain of team games, with the example of rugby [13].

The aim of our research project is to realize a KBS for analysing game sequences from video image sequences of a given team, in order to propose tactical improvements during the training stages for the next matches [14]. Such a system is to be seen as a software tool to assist tactical decision-making. Its definition has been preceded by a domain formalization step, and a reasoning model based on a typical game sequence partition, and on different inference levels [12].

The interpretation is carried out by the expert on symbolic - and no longer video - image sequences. Such sequences are reconstructed from the data resulting of an image processing module, coming from the input colour video images. Such a module is located upstream of the visual knowledge expression software [15].

Downstream our software is interfaced with a complementary 3D image animation module in order to simulate at an individual level all the tactical solutions proposed by the system.

3.1 Visual knowledge acquisition and representation languages

Diagramming languages and spatial reasoning

Casner has proposed an interesting contribution to the problematics of Problem Solving Methods for Visual Knowledge processing, by formally defining, then using in separate domains, diagramming languages [6]. Such languages, that could be seen as particular languages for Visual Knowledge Acquisition, differ from iconic languages because they are defined as strict mathematical applications between a set of perceptual codes (such as patterns, colors, or spatial combinations of isolated objects) and a set of interpretations, whose goal is to associate a semantic level within the use of these perceptual codes. By choosing team games, and more precisely american football, for applicative domain, Casner has generated the corresponding perceptual codes with their interpretation for each considered domain. The resulting system, called BOZ, enables to define and implement diagramming-based convivial conventions.

The diagramming language use iconic representations, spatial distributions and graphical symbols to represent the entities at an individual level (players), a collective level (teams, attack, defense) and a meta-collective level (global movements, strategies).
Such diagrams refer to the associated concepts.

So the BOZ system is a generic software tool for defining and using diagramming conventions, which can be seen as specific Human-System Cooperation languages. A formal definition is proposed both in terms of perceptual codes (that could be include in any interface) and of interpretations associated in manipulating such diagrams by the expert. Otherwise stated a complete set of diagramming conventions can be given in an extended BNF formalism. This language allows the expert to define interactively new codes, interpretations, and conventions

Finally the implementation has been processed by following an object-oriented specification and implementation methodology : the objects and associated functions are Lisp written.

Let us now intoduce a fundamental contribution in the field of VKA.

Interactive visual languages for knowledge acquisition

Gaines has proposed an interactive visual language both for the acquisition, representation, and editing of knowledge structures for term subsumption languages [9]. This language is formal in that way it is completely defined at the lexico-syntactic and semantic levels. Besides it intertranslates with textual knowledge represen-tation

languages. Such a language is supported by an interactive graphic structure editor, which offers an easy and natural way of cooperating with the user.

In the same way that Casner, a visual language is defined equivalent in expressive power to term subsumption languages expressed in a textual form. At any basic knowledge representation function is associated a visual form expressing it precisely and completely.

An important point is the following equivalence : the visual and textual languages are intertranslatable. Expressions in the language are graphs composed of labeled nodes and directed (or not) arcs, just like in the semantic networks formalism. The nodes are labeled textually or iconically, and their types are denoted by several distinct outlines.

In the corresponding system architecture, a part of the knowledge acquisition toolkit is immediately associated with the structure editor. At the same time, this editor is coupled to a knowledge representation server [10]. It can export visual data to several Expert System shells, and to the textual language of the knowledge representation server. The server also imports knowledge structures from other tools, such as the KSSO repertory grids elicitation and induction module. So this visual language is interactive, and dedicated to knowledge acquisition. Finally note that the same approach has been used for problem solving methods, in the case of the Sisyphus example [8].

Let us now study our VKE contribution, related to the two previous languages.

3.2 Visual Knowledge Elicitation

Choosing the experts

Our approach, which is focused on knowledge acquisition and representation, integrates Pierre Villepreux's expertise, high level competition player then famous coach and technical director, and considered by other coaches as an high-level game theoretician [17].

Our choice is explained by the theoretical formalization level reached by the expert and by the existence of a more general model, consistent with this expertise and which is necessary to understand it [7]. Proposed by René Deleplace, such an including formal framework is the so-called "in action Tactical Choice Systematics" or "logical deleplacian model", which is as well theoretical as practical.

Integrating the VKE language into a KA software tool

Our Visual Knowledge Elicitation language is integrated to a specific visual KA software tool.

Its software specifications follows an object-oriented conception and programming methodology, and the supported language is C++ under SunView and X Windows environments unified by OpenWindows. On the hardware point of view, it runs on a Colour Sun 4 Workstation (SPARC station IPC).

The Visual Knowledge Elicitation language is based on selecting isolated and global objects (discrete or not), in order to represent pertinent subsets of the considered symbolic picture. Besides the design of objects, the expert can give the trajectory of the manipulated subsets in terms of parallel or sequential movements. Thus owing to an explicit selection mechanism, here implemented by a mouse, he designs the objects-concepts he manipulates during his reasoning.

Thus we can associate visual information - which can be approximated by quite simple geometrical models - to reasoning sequences, and movement expression characteristics, that the today's help tools for acquisition and structuration do not allow to take into account or to backup in order to design more and more efficient KBs. This elicitation language is above all domain independent.

3.3 Static and dynamic formal definition of the VKE language

Formalism and associated notations

The next section describes the VKE language FG using the BNF formalism which is defined by two operators and four constructors as follows :
- disjunction noted " | " : A ::= B | C means : "B or C are derived from A",
- conjunction noted "." : A ::= B . C means : "B then C are derived from A",
- Kleene's closure noted "*" : A^* means : "from zero up to n times A (n integer > 0)",
- positive closure noted "+" : A^+ means : "from once up to n times A (n integer > 1)",
- empty string noted "ε" : analogous to the empty set in set theory, is a character string of length zero,
- omission noted "[ ]" : [A] means : A | ε.

As stated before, the BNF formalism is a particular context-free grammar. Widely used nowadays in formal specification of languages, its notations are the following :
- the meta-linguistic variables are written between "<" and ">" separators,
- the terminal symbols are written with no special separator,
- the "::=" symbol means "is composed of" : it is the well-known derivation operator,
- the disjunction, noted " | ", is only used in the right hand side of the production rules, because we manipulate here context-free grammars,
- the conjunction is implicit and associated to the usual writing order (left to right) of terminal and non-terminal symbols.

Stages in the formal definition of the VKE language

The actual language design process using the combined BNF and RE formalisms, from formal specification to use, has these complementary parts : initial static definition, then dynamic description at use by the expert. Thus to turn specification into action, we have to follow two steps.

Static definition of the VKE language by a FG

The BNF formalism is seen as a Knowledge Representation formal language. By combining them, we give the BNF productions of our static VKE language.

The associated FG is composed of a finite set of productions (its cardinality is 16), and starts from the <VK ELICITATION> axiom. In the following description, terminal symbols are indicated in lowercase, non-terminals between brackets. The rules are numbered for convenience of reference. No ordering is in fact implied. Headings have been inserted to facilitate locating sections of the grammar. The resulting sections are to be used as a rough guide only.

Terminal symbols : any string in lowercase,
Non-terminal symbols : any variable between inferior and superior meta-symbols,
Start symbol : VK ELICITATION,
Meta-symbols : BNF constructors and operators,
Production rules :

Start symbol
1. <VK ELICITATION> ::= (<Space-Time> . <Movement>)$^+$

Space and time
2. <Space-Time> ::= [<Object> trajectory]

Movement definition
3. <Movement> ::= <Space-Time> <Relative speed> <Sequentiality>

Object definitions
4. <Object> ::= <Isolated object> | <Global object> . <Shape name>
5. <Isolated object> ::= visual representation of an elementary entity {ball, player}
6. <Global object> ::= <Discrete global object> | <Non-discrete global object>
7. <Discrete global object> ::= a visually countable set of isolated objects
8. <Non-discrete global object> ::= a visually non countable set of isolated objects

Shape selection and description
9. <Shape name> ::= name given by the expert to the global object shapes he has drawn

Speeds
10. <Relative speed> ::= <Less fast speed> | <Equal speed> | <Faster speed>
11. <Less fast speed> ::= first option of "ad-hoc" visual conventions
12. <Equal speed> ::= second option of "ad-hoc" visual conventions
13. <Faster speed> ::= third option of "ad-hoc" visual conventions

Displacements

14. <Sequentiality> ::= <Sequential movement> | <Parallel movement>
15. <Sequential movement> ::= sequential in relation to the last manipulated object
16. <Parallel movement> ::= parallel in relation to the last considered object

Dynamic definition at use of the VKE language by a FG

During their use by the expert, there is a running order for the commands. Such an order is defined by the following FG given in the BNF formalism. This dynamic VKE language FG is composed of a set of p production rules (p = 15). In the same way, the rules are numbered for convenience of reference, but no ordering is implied. Headings have been inserted to facilitate locating sections of the grammar.

Terminal symbols : any string in lowercase,
Non-terminal symbols : any variable between inferior and superior meta-symbols,
Start symbol : FUNCTIONS,
Meta-symbols : BNF constructors and operators,
Production rules :

Start symbol

1. <FUNCTIONS> ::= (<Initialising> . <Processing> . <Finishing up>)$^+$

Initialisation step

2. <Initialising> ::= <Filename> . <Acquisition start>
3. <Filename> ::= the coordinates filename corresponding to the current image
4. <Acquisition start> ::= reads the coordinates file, shows the initial spatial distribution

Ending step

5. <Finishing up> ::= <Session writing> . <End of acquisition>
6. <Session writing> ::= file writing of the expert's work session
7. <End acquisition> ::= quit the KA software tool

Working sessions

8. <Processing> ::= (<VK ELICITATION> | <Chrono> | <Match> | <P-Simulation> | <S-Simulation> | <D-Modelling>)*
9. <VK ELICITATION> ::= see the previous section
10. <Chrono> ::= time quantification in min, sec and dix, then visualise next image
11. <Match> ::= team names, competition and context {day, year, and town}

Parallel and sequential animations

12. <P-Simulation> ::= parallel movement of all the objects whose trajectory has been explicitly given by the expert
13. <S-Simulation> ::= sequential movement of all the objects whose trajectory has been explicitly given by the expert

Deep knowledge modelling

14. <D-Modelling> ::= [<Inflexion>] | (<Inflexion> . <Inflexion>)$^+$
15. <Inflexion> ::= visual representation, according to Deleplace's theory of the speed vector and inflexion limits; eventually visualisation of these basic concepts in hidden mode, because there are not always useful during the expert's work.

3.4 Acquisition sessions with the expert

Users interface

Moreover discrete and non discrete objects, relative speeds, sequential or parallel displacements, other functions have been implemented. First the "Start-Acquisition" and "End-of-Acquisition" options, which respectively initialise the expert's work by reading the input data file (2D players coordinates) and quit the software.

The semantic difference between the VKE language and the complementary functionalities is done on the interface itself, composed of three parts :

- an *horizontal commands area*, located up and down the screen, and including the functions (up, the processing options and down, the filename associated to the picture),
- a *vertical commands area*, located on the right side of the screen, and dedicated to the options of the VKE language,
- the *central commands area* contains an aerial symbolic representation composed of the static background "seen from a bird eye", including all the (2p+1) individual actors i.e. the p attackers, the p defenders and the ball.

Analytic geometrical models for pattern approximation

The expert can draw discrete or non discrete global objects, which represent convex or concave patterns given by their closure. Any closure is defined by its polygonal approximation, seen as a set of segments. Our aim is to find in this case the most suitable pattern model of this drawn on the screen by the expert. There is no restriction about the patterns that could be designed : the expert can indeed propose the pattern he wants when solving a problem.

Such an approach is of a double interest. On the one hand for the expert, in order to help him to describe and formalize the patterns he uses in his resolution models by simple geometrical figures. On the other hand for the knowledge engineer, in order to capture non ambiguous objects for the knowledge representation and structuration phase.

We want to provide the expert with a generic geometrical palette. There are a large number of analytic models. Nevertheless for our domain the following classes hierarchy is sufficient : conics : the ellipse and circle models, parallelograms : the rectangle and square models, polygons : particularly the n-side regular polygon.

Domain dependent functions

The three domain dependent functions are the following :
- context (match and competition framework),
- time (min, sec and dix) associated to the current image sequence owing to the "Chrono" button,
- for any isolated object considered and taking into account its specific individual role (attacker or defender), the Deleplace button is dedicated to visualize the concepts coming from the deep knowledge i.e. the deleplacian modelling of general movement phases : module and direction of the speed vector, the inflexion angle generating the inflexion cone. These informations are shown in transparent mode again if the button is reactivated since there are not useful at any time when the experts infers and proposes his tactical solutions.

Handling sessions

The following screen copy is an image of a current visual knowledge acquisition session. We can see that a geometrical pattern model palette is proposed to the expert on the right side : it is composed of the first four figures mentioned in the last previous section : ellipse, circle, rectangle and square. After drawing any pattern, the expert wants to approximate it by one of these options in choosing the associated button : the software tool ajusts then automatically the model to the drawn pattern. Indeed a program is able to do this task by itself in giving the minimum distance model via a surface calculus. This avoids the expert tricky or even unexpected handlings.

Interactive simulation of the reasoning steps

The "Sequential-Animation" and "Parallel-Animation" buttons allow respectively to visualize at any time and in real-time a behavioural simulation of the players displace-ments only for these whose trajectory has been explicitly given by the expert. There are two options. If the action is sequential, the software makes a complete displacement of the considered players one after another. In case of simultaneous trajectories, players are all animated in parallel step by step.

The expert proceeds to such simulations either after each reasoning substage, or at the end of the working session. The visual trail of each simulation is printed on a paper sheet, by using interactive black-and-white screen hardcopies, which are both kept by the expert and the knowledge engineer.

Backup of visual informations

Let us first make a difference between record and backup. The record is a complete file writing from the beginning to end of the session. The backup is the exact copy of the

current state of the KA session. Otherwise stated, the record is the union - in sense of the set theory - of the all the backups from the initial to the final states.

Keeping in mind the knowledge representation phase for the KB structuration, we want to have a trail of the acquisition work with the expert. So a Session-Backup option is proposed and enables to write on the disk all the informations associated with isolated or global objects. Concerning the special "Ball" object, if a player is keeping it, the algorithm consists in duplicating the data. For discrete or non discrete global objects, the same procedure is executed. Indeed even if the visual counting of punctual objects is not possible, these objects are in fact present within the symbolic representation.

An optimization would consists in recording (and not making a backup any more) the manipulated visual knowledge. But the major disadvantage would be the very large amount of redundant data to be written on the disk.

4. Non verbal knowledge representation and structuration

4.1 Why a structuration stage is essential

For understanding pictures, defining and implementing a VKE language is not sufficient. Indeed we have to consider a knowledge representation and structuration phase in order to design the KB. This is done by using the MACAO methodology and running the associated software tool [3]. More precisely for any problem solved by the expert, by using :

- the semantic networks formalism for domain modelling,
- the schemes formalism for reasoning modelling.

4.2 The MACAO knowledge modelling methodology and tool

MACAO is a knowledge elicitation and modelling methodology which has two generality levels i.e. is both domain and expert independent [2]. A well-known decomposition of expertise transfer for KBs construction distinguishes for the data, their elicitation, abstraction, analysis, structuration then validation.

The major drawback of the elicitation stage consists in making explicit knowledge sources, which are by convention implicit et compiled via an universal communication media as natural language.

The following steps concern two different problems we have to distinguish :

- information analysis, in order to separate the concepts, their structural relationships or the relations coming from the problem solving method itself,
- formalism transformation, in order to turn the initial data in natural language into a set of intermediate representations to achieve coding in a target programming language.

The aim of this KA methodology is to assist the whole expertise transfer process, to design and refine the global Conceptual Model. This model plays 4 roles in MACAO :

- i) filter between natural language and software final code : it allows to search representative reasoning objects, and helps to capture the essential expertise information and the manner of how there are used along the usual "interpretation, decision, and action" cycle,
- ii) descriptive framework of knowledge for formalizing the information transformational process,
- iii) access key to a given know-how : in order to translate the complete knowledge, it allows accessing specialists know-how,
- iv) fixed framework dedicated to the knowledge engineer, to minimise the variations of the behavioural analysis and the experts verbalisations during the acquisition step.

So this typical bottom-up acquisition methodology enables to construct a Conceptual Model by knowledge elicitation via solving all major problem classes.

4.3 How the verbal knowledge structuration is processed with MACAO : static and dynamic knowledge representation

The dynamic knowledge associated to the reasoning is expressed with MACAO by using schemes graphs. Such a structuration stage is induced by that of knowledge acquisition and linked to the schemes representation language [4].

In the current version, schemes include different text fields, to be filled when solving a problem :

- text-name : quick specification of what is supposed to do the scheme,
- text-parameters : input and output parameters are distinguished,
- text-context : defines the context when the scheme considered is called,
- text-goal : the purpose of the scheme when it is called,
- text-strategy : which strategy is used in the scheme,
- processing : the set of procedures and next schemes called by this scheme with the five associated operators :

  conjunction : ET, succession : PUIS, discrete choice : CAS,
  disjunction : OU, iteration : TANT QUE.

  Note that the processing field is empty if and only if the considered scheme is terminal in the resolution graph.

We could give an example of resolution for solving the problem "Attaque-Jeu-Déployé". Two graphs show respectively the static knowledge and dynamic knowledge modelling, i.e. the semantic nets for domain modelling, and the schemes for reasoning modelling.

4.4 Extending MACAO for taking into account visual data

Description of the extension

Concerning the representation phase, we have to extend the scheme structure in order to take into consideration the visual knowledge manipulated within the expert inferences. Such an extension is done through inserting a link on the graphic visualization (characteristic pattern on a symbolic image) corresponding to all the specific concepts manipulated in the scheme. This insertion implies to update the existing structure with including a logical link in the CONTEXT and GOAL fields, by adding a new "DISPLAY IMAGE" button, in order to show respectively the input defensive and the output offensive spatial distributions, perceived thanks to the corresponding patterns.

Note that an equivalent processing could be done by updating only one field, that of PARAMETERS, with inserting in the input parameters the input defensive distribution image and in the output parameters the proposed output offensive simulation picture. These insertions are implemented by including logical links to the associated images.

The consequence of such an extension consists in creating two symbolic picture directories, structured by typical defense and attack distribution patterns, following both inference levels (Individual, Line and Total Collective levels for the considered domain) and game types (Static, Semi-Dynamic and Dynamic sequences).

Generalising our KA approach to related expertise domains

It seems obvious that we could easily transpose our approach to the growing number of experienced tasks which include nowadays a picture understanding activity, on either fixed images or image sequences. There are plenty of applicative domains, such as static or dynamic imagery in sportive, medical, satellite or technical areas.

Indeed our technique for elicitating non verbal expertises is generic and could be fisrt applied to other team games, because the status of the manipulated objects remain exactly the same for isolated players, sets of players and trajectories. No significant changes arise in the visual acquisition technique, and the same methodology could be run easily with a football or basketball high-level expert.

5 Conclusion

The originality of our approach consists of the proposed VKE language. Indeed our visual KA methodology, for non verbal expertises, is based on this generic VKE language, which has been defined precisely, and whose use has been shown in the domain of team games. Its basic interest is that it is rather suitable to the specific needs in picture understanding than the usual verbal methods, all starting from written reports.

Let us now compare it with Casner diagramming language, and above all with Gaines interactive visual language. All are seen as formal languages, and so have been formally defined syntactically and semantically. Like Casner, we use the same (extended or not) BNF formalism, but Gaines prefers to use the CLASSIC expressions. All are full interactive and convivial for the user. In the three languages, interpretations through concept names are associated to the visual entities. The difference is that the Gaines inter-translatability between the visual and textual languages is here not respected. Another difference is that Gaines model is finite, but our language is infinite. It is indeed related to a pattern understanding activity. These patterns come from the domain configurations, and their number is a priori infinite. Finally the same object-oriented specification and programming methdology is used. Only the support language is different.

References

1. P. Albert, I. Bousquet, E. Brunet, G. Jacques. *KADS-TOOL, an industrial work-bench for the KADS methodology.* Proceedings of the 5th International Conference on Software Engineering, pp 145-154, Toulouse, France, December 1992.

2. N. Aussenac *Conception d'une méthodologie et d'un outil d'acquisition de connaissances expertes.* Ph.D. Thesis of the Paul Sabatier University, Toulouse, France, 1989.

3. N. Aussenac, J. Frontin, J.L. Soubie. *Evolution d'une représentation des connaissances pour l'acquisition.* Proceedings of the International Conference on Knowledge Modelling and Expertise Transfer (KMET'91), pp 135-148, Sophia-Antipolis, France, April 22-24, 1991.

4. N. Aussenac, N. Matta. *Making the method of problem solving explicit with MACAO : the Sisyphus case-study.* Proceedings of Sisyphus'92, M. Linster (Ed.), GMD Internal Report No 630, Bonn, Germany, March 1992.

5. J.H. Boose. *A survey of knowledge acquisition techniques and tools.* Knowledge Acquisition, Vol 1 No 1, pp 3-39, Academic Press, March 1989.

6. S. Casner. *Building customized diagramming languages.* in Visual Languages and Visual Programming, pp 71-95, Plenum Press, NY, Chang (ed.), 1990.

7. R. Deleplace. *Rugby de Mouvement, Rugby Total.* EPS Editions, Paris, 1979.

8. B.R. Gaines. *The Sisyphus problem-solving example through a visual language with KL-ONE like knowledge representation.* Proceedings of Sisyphus'91, M. Linster (Ed.), GMD Internal Report No 663, Bonn, Germany, July 1992.

9. B.R. Gaines. *An Interactive Visual Language for Term Subsumption Languages.* Proceedings of the Twelfth International Conference on Artificial Intelligence (IJCAI'91), Volume 2, pp 813-823, Sydney, Australia, August 24-30, 1991.

10. B.R. Gaines. *Empirical investigation of knowledge representation servers : design issues and applications experience with KRS.* Proceedings of the AAAI Spring Symposium on Implemented Knowledge Representation and Reasoning Systems, pp 87-101, Stanford, California, 1991.

11. J.P. Krivine, J.M. David. *L'acquisition des connaissances vue comme un processus de modélisation : méthodes et outils.* Intellectica Review, No 12 pp 101-137, Paris, December 1991.

12. B. Singer, P. Villepreux. *Vers une formalisation de l'analyse sémantique de matches en sports collectifs. Application au rugby à XV. Towards formalizing the semantic analysis of team games with the example of rugby.* International Review Mathématiques Informatique Sciences Humaines No 114 pp 19-33, Paris, Summer 1991

13. B. Singer, P. Villepreux, P. Dalle, F. Baucher. *Le système RUGBY, ou Comment il aurait fallu jouer.* Proceedings of the Journées Internationales d'Automne de l'ACAPS (ACAPS'91), pp 118-119, Lille, France, 9-11 November 1991.
14. B. Singer, J.L. Soubie, P. Villepreux. *L'Intelligence Artificielle au service du rugby : construction d'une Base de Connaissances de prise de décision tactique.* Proceedings of the International Conference on Software Tools : Applications to Sport and Physical Education (STASPE'92), pp 21-36bis, Paris, France, May 1992.
15. B. Singer, P. Villepreux, J.L. Soubie. *Intelligence Artificielle et sports collectifs : une Base de Connaissances d'aide à la prise de décision tactique.* Proceedings of the First International Colloquium on Sports and Computer Science, Artificial Intelligence and Expert Systems Day, INSEP, Paris, January 20-22, 1993.
16. L. Steels. *Components of Expertise.* AI Magazine, Vol 11 No 2, pp 28-49, Summer 1990.
17. P. Villepreux. *Rugby de mouvement et disponibilité du joueur.* Mémoire en vue de l'obtention du diplôme de l'INSEP, Paris, 1987.
18. B.J. Wielinga, A.T. Schreiber, J.A. Breuker. *KADS : a modelling approach to knowledge engineering.* Knowledge Acquisition, Special Issue : The KADS approach to knowledge engineering, Vol 4 No1, pp 5-53, March 1992.

A visual display system for the teaching of intonation to deaf persons: A preliminary report

Gerard W.G. Spaai[1,2], Esther S. Derksen[2] and Paul A.P. Kaufholz[1,2]

[1]Institute for Perception Research/IPO, Eindhoven, The Netherlands
[2]Institute for the Deaf/IvD, Sint-Michielsgestel, The Netherlands

Abstract. To help profoundly deaf speakers improve their intonation, a visual display system for teaching intonation is being developed. In this system, the so-called Intonation Meter, visual feedback of intonation is given as a continuous representation of the pitch contour containing only the perceptually relevant aspects of the intonation pattern. An explorative study was carried out to determine the usability of the Intonation Meter for teaching intonation. Preliminary data collected with a small number of profoundly deaf children, indicate that the system can be an important instructional tool for teaching intonation. Suggestions for increasing the usability of the system are given.

Keywords. Speech training devices, visual feedback of intonation, instructional technology, GUI.

1 Introduction

Children who are born with little or no functional hearing capacities or who acquire a severe functional hearing loss before the age of about one year and six months, i.e. before any substantial speech or language development could take place, do not acquire speech skills spontaneously. They have to develop these skills by the explicit use of mainly visual information and reliance on sensorimotor control in addition to residual hearing capacities. Although it is well known that some prelingually, profoundly deaf children are able to acquire adequate speech skills under conditions of prolonged individualized speech training with a speech therapist (Nickerson & Stevens, 1973) others have problems producing speech that is fairly intelligible to the ordinary listener.

There are two main reasons for the problems deaf children have in developing intelligible speech. First, deaf children do not learn to relate the articulatory processes with the corresponding acoustical signals and do not learn to monitor their articulation with audition (Tye-Murray, 1992). Because deaf children have limited or no access at all to acoustic speech targets for comparison with their speech products, deficiencies in the segmental and suprasegmental aspects of speech are likely to occur (Gold, 1980; Hudgins & Numbers, 1942; Nickerson, 1975). Second, only less than half of the speech sounds can be recognized by vision. Thus, deaf children receive an incomplete model of the

repertoire of speech sounds to be mastered.

Deficiencies in the segmental aspects of speech often include the neutralization of vowels due to insufficient variation in the second formant as a result of centralized tongue positions with limited amounts of movements while producing the different vowels (Angelocci, Kopp & Holbrook, 1964; Dagenais & Critz-Crosby, 1992), and distortion, submission and omission of consonants (Hudgins & Numbers, 1942; McGarr & Osberger, 1978). Furthermore, compound consonants and diphthongs are often too slowly articulated resulting in the addition of one or more vowels.

Deficiencies in the suprasegmental aspects of speech often involve a poor timing of speech. Specifically, deaf speakers often speak at a slower rate than hearing speakers (Nickerson, 1975) and have a poorer speech rhythm (Hudgins, 1946). They also often have problems with sounds requiring precise coordination of the timing of articulatory movements and rapid transitions of one articulatory target (position) to another. Other suprasegmental errors involve problems in breath control and control of prosody such as intonation, stress and emphasis. The characteristic difficulties related to the control of prosody include abnormally high pitch (Angelocci, Kopp & Holbrook, 1964; Stathopoulos, Duchan, Sonnenmeier & Bruce, 1986), excessive phoneme related variations in pitch (Bush, 1981) and a lack of linguistically relevant pitch variations. Better production of pitch in the speech of deaf persons is important because it contributes to speech quality and speech intelligibility especially when the segmental aspects of speech are produced fairly well (Metz, Schiavetti, Samar & Sitler, 1990).

2 Visual feedback of intonation

It is difficult for deaf speakers to learn to control the pitch of speech because: (1) they may not have sufficient residual hearing to perceive the auditory cues for control of pitch; (2) tactile and proprioceptive feedback play only a minor role in pitch control (Ladefoged, 1967), and (3) the major variations in pitch are determined by the action of the cricothyroid muscle (Collier, 1975), which means that it is impossible to give visual cues on the *control* of pitch. Therefore, to help deaf speakers acquire better pitch control, various researchers have developed sensory aids that extract the pitch from speech and display it visually. The main reason for developing such aids is that they are capable of providing objective and immediate feedback on pitch. Teaching intonation with the help of such a display will only be productive if its use results in the establishing of effective speech motor activity. Deaf speakers will acquire this from learning the speech motor actions they must perform in order to imitate a visually displayed intonation pattern correctly.

Little is known about the effectiveness of visual intonation-display systems for teaching intonation to profoundly deaf persons. Evaluations of these systems (Abberton, Parker & Fourcin, 1978; Friedman, 1985) are often based on case studies which were descriptive and not experimental in nature thereby allowing for only restricted statements concerning the effectiveness of these systems. Other studies (e.g., McGarr, Head, Friedman, Behrman & Youdelman, 1986; McGarr, Youdelman & Head, 1989) incorporated the use of a visual intonation-display system into an experimental speech

training program. The effects of this experimental condition were then compared to a control condition in which a regular speech training program had been used. Practice conditions then differed in more than one respect making it difficult to determine the effectiveness of the visual intonation-display system. Finally, the usability of these systems was often determined with students having severe hearing losses (Youdelman, MacEachron & Behrman, 1988; Youdelman, MacEachron & McGarr, 1989). It is unknown, however, whether the findings of these studies are generalizable to profoundly deaf persons.

In summary, it is concluded that the usability of visual intonation-display systems for teaching intonation to profoundly deaf persons remains to be explored fully. This, may have contributed to the lack of widespread use of these systems in schools. Another factor that may have contributed to this is that in these systems pitch was measured and directly fed back to the deaf speaker without post-processing the pitch contour. Two difficulties arise when speakers receive the unprocessed pitch contour.

The first problem arises from the fact that the interpretation of the displayed intonation contour is hampered by the interruptions during unvoiced parts which are at variance with the continuously perceived pitch contour. The second problem relates to the presence of many perceptually irrelevant pitch variations (the so-called microintonation) in unprocessed pitch contours which may distract the attention from the perceptually relevant pitch variations. Microintonation can be very conspicuous, especially at transitions between consonants and vowels. It can barely be perceived, if at all, by persons with normal hearing, let alone be imitated. In order to solve these problems, a system has been developed, the so-called Intonation Meter (Spaai, Storm & Hermes, 1993), that gives visual feedback of intonation as a continuous representation containing only the perceptually relevant pitch variations. That is, the course of the pitch contour is approximated by a small number of straight lines resulting in a stylized pitch contour. This pitch contour representation is supposed to facilitate the interpretation of the visual feedback of the intonation contour. An example is shown in Figure 1 for the Dutch sentence "Op een dag kwam een vreemdeling het dorp binnenwand'len" (One day a stranger came walking into the village). The separate dots show the unprocessed pitch measurements and the continuous straight lines show the stylized pitch contour.

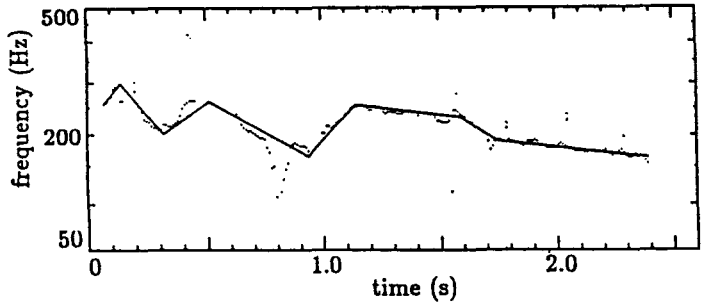


Figure 1. Display of the unprocessed pitch measurements and the stylized pitch contour for the Dutch sentence "Op een dag kwam een vreemdeling het dorp binnenwand'len" (One day a stranger came walking into the village).

The following section presents an explorative study that was carried out to find out whether the visual intonation-display system, i.e. the Intonation Meter, can be helpful for teaching intonation to profoundly deaf persons. In other words, the question is addressed whether profoundly deaf children who receive intonation training that incorporates the Intonation Meter as part of the training program show greater improvement in intonation skill than a control group that received intonation training without the visual display system. In the final section suggestions for increasing the usability of the system are given in terms of the development of a graphical user interface.

3 Experiment

Design and Procedure

Two groups of profoundly deaf students practised intonation. The first group, the *control group*, practised intonation with the help of regular means. This group received speech training using mainly auditory input via the child's personal hearing aids. In addition, information on pitch was for instance presented via some visual activity of the speech therapist or via a representation on paper. No visual or tactile sensory aid was used with this group. The second group, the *experimental group*, received speech training using auditory input and the Intonation Meter to provide visual feedback on pitch. Typically, when a student is working with the system, a pitch contour example produced by the speech therapist is displayed on the upper part of the screen of the Intonation Meter which must be imitated on the lower part. While imitating, direct unprocessed feedback is given by means of real-time pitch measurements. After conclusion of the whole utterance the stylized contour is displayed too. The learning progress of both groups was compared to a third group, the *test group*, which did not receive any extra practice on intonation.

Learning progress was measured by differences on an intonation test that was administered prior to intonation training and also at the end of the training period, the so-called pretest and posttest. This intonation test was in part based on the Fundamental Speech Skills Test (FSST). The FSST (Levitt, Youdelman & Head, 1990) assesses suprasegmental production, e.g., production of pitch contours in words and sentences. For the purpose of the experiment, the FSST was translated and adapted to the Dutch language. Furthermore, the production of pitch variations in long vowels and sequences of syllables was assessed.

The intonation test was administered without sensory aids except for the students' personal hearing aids. The students' productions were tape-recorded and a speech therapist who was experienced in listening to and evaluating the speech of deaf children rated the recordings. For the purpose of this explorative study results are reported for: (1) the production of appropriate average pitch in words and sentences, and (2) the production of pitch variations in long vowels and syllables.

Training Sessions

The experimental group and the control group practised intonation three times a week, each session lasting about 15 minutes. This was done over a four months period. Thus, in all, each student in the control condition and the experimental condition received about

eight hours of intonation training. The students participated in this 'experimental speech training program' in lieu of receiving the regular speech-training regimen.

Generally speaking, speech training focused on the remediation of an inappropriate average pitch and, although to a less extent, on the production of pitch variations in vowels and syllables. Practice consisted of four groups of activities: awareness training, auditory discrimination and identification, imitation and production-on-demand activities. In pitch awareness activities, attention was focused on the concept of pitch, e.g., what do high or low refer to. In auditory identification and auditory imitation activities the attention was focused on learning to discriminate and identify pitch (variations) through auditory cues. Imitation involved the imitation of a pitch contour produced by the speech therapist whereas production-on-demand required the production of pitch contours without benefit of the teacher's model. The four groups of activities were not necessarily hierarchical and training was generally performed in several of these areas simultaneously as recommended by McGarr, Youdelman and Head (1992). Pitch contours were trained on sustained vowels, syllables and words.

Subjects

Twelve prelingually, profoundly deaf students from a secondary school for special education at the Institute for the Deaf in Sint-Michielsgestel, The Netherlands, participated in the experiment. All students received speech training prior to the experiment. They had been educated according to the 'oral reflective method' which is basically an oral-aural approach to speech training (Van Uden, 1977). Ages of the subjects ranged from 14 to 20 years. They had no motor or intellectual handicaps. All students had a profound hearing loss greater than 90 dB ISO bilaterally. They used acoustic hearing aids binaurally and produced pitch contours that were perceived by their speech therapists to be "relatively flat" reflecting a monotonous voice. In other words, the subjects were unable to produce linguistically relevant pitch variations. Also, in some cases, their pitch appeared to be too high with respect to age and sex. Subjects were matched in groups of three as closely as possible according to their age, residual hearing, academic performance and speech skills. After the subjects had been matched into three groups attrition took place. The numbers of children participating in the control condition, the experimental condition and the test condition were three, four and four, respectively.

Results

Average Pitch. Figure 2 shows the percentages of acceptable ratings at the pretest and the posttest for the average pitch in isolated words and sentences. The data are presented for the experimental group (4 subjects x 34 ratings = 136 judgments), the control group (3 subjects x 34 ratings = 102 judgments) and the test group (4 subjects x 34 ratings = 136 judgments). Improvement was measured by a shift from an unacceptable rating at the pretest to an acceptable rating at the posttest. The percentage of acceptable judgments for average pitch increased for all groups. However, it is immediately apparent that the experimental group made more progress than the test group and the control group. Furthermore, posttest results illustrate that the control groups performed only slightly better than the test group.

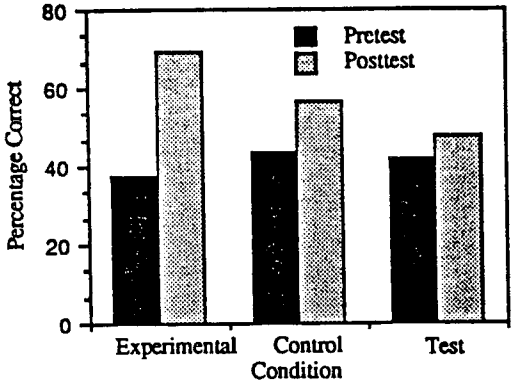


Figure 2. Average pitch ratings. Percentages of acceptable judgments are plotted for the experimental group, the control group and the test group. The appropriateness of average pitch in isolated words (22) and sentences (12) has been rated. The data are presented for the pretest and the posttest.

Pitch Variations in Vowels and Syllables. Figure 3 shows the percentages of acceptable ratings at the pretest and the posttest for the production of pitch variations in long vowels and syllables. The data are presented for the experimental group (4 subjects x 153 ratings = 612 judgments), the control group (3 subjects x 153 ratings = 459 judgments) and the test group (4 subjects x 153 ratings = 612 judgments). The results clearly show that the experimental group made most progress (64%). The control group and the test group also showed progress but the magnitude of their improvement was not as large: the percentage of acceptable judgments increased by about 17%.

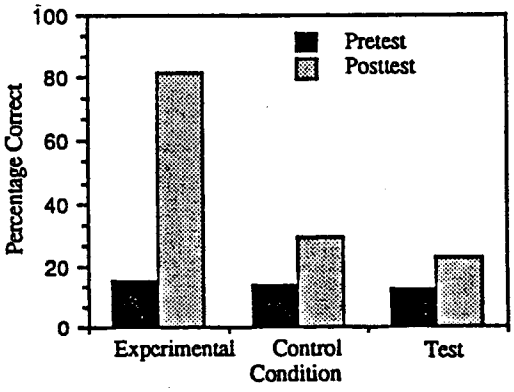


Figure 3. Average ratings of the production of pitch variations in vowels and syllables. Percentages of acceptable judgments are plotted for the experimental group, the control group and the test group. The appropriateness of the production of pitch variations in vowels and syllables has been rated. The data are presented for the pretest and the posttest.

Discussion

The results of this explorative study showed that profoundly deaf students who received intonation training that incorporates the use of the Intonation Meter show greater progress in the production of appropriate average pitch and the production of pitch variations in vowels and syllables, than a control group who received regular training. Although these results were gathered with a small number of children which may make it difficult to generalize, they are very promising especially since training time was limited. Also, the children participating in this study could be characterized as having more impervious phonatory problems owing to their age. Therefore, research is in progress to find out whether speech training that incorporates the use of the Intonation Meter can be more effective with young children. Long term research is necessary to determine whether the intensive use of this system can result in improved control of intonation in connected discourse.

4 Future Developments

Graphical User Interface

The Intonation Meter is meant to be used for teaching intonation to deaf children aged 4 to 20 years. Since the students and the teachers are not expected to have any computer experience, the ease of use is extremely important. A future version of the Intonation Meter should support two modes, a *Teacher mode* and a *Student mode*. In the Teacher mode a speech therapist is monitoring the deaf student. Here, the speech therapist can adjust parameters specific for an individual student, select exercises and give instructions. In the Student mode, the student is practising intonation without the presence of a speech therapist. By merely entering his or her name a student can start a training session. The set-up of all system parameters is done automatically and the student is guided by the Intonation Meter. Parameter settings and selection functions are not accessible for the student. Records of performance are kept for each individual student to provide computerized guidance.

The user interface in the Teacher mode will differ from the user interface in the Student mode in that the accessibility of the program functions is essentially different. That is, in the Teacher mode all program functions should be made accessible whereas in the Student mode only essential functions, e.g., starting and stopping, are accessible.

To meet these requirements two parts of data presentation in the user interface, have to be designed carefully. First, the measured data is represented in the so-called View. Second, the organization of functions and features is kept in the Dialogue structure. The interpretation of the display depends on the attributes of the graphical representations of the measured speech characteristics, e.g., line thickness for amplitude. These graphic representations must be clear and easy to interpret. In addition to these graphical representations, fast feedback of measured data is essential. The dialogue structure implies the organization of the functions and features in dialogue boxes. This part of the user interface is actually different for both modes because the accessibility to functions is defined here. An overview of the two parts of data presentation is presented in Figure 4.

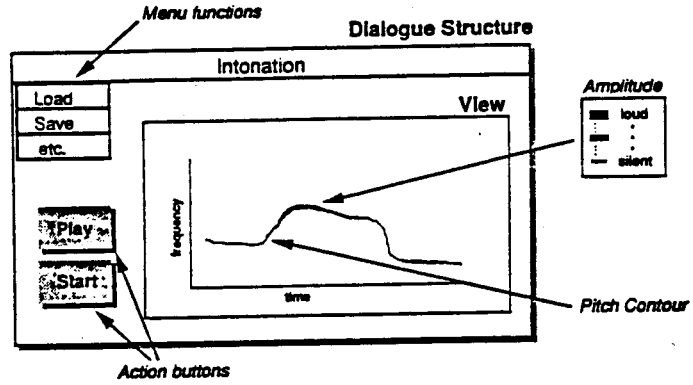


Figure 4. Schematic concept of the graphical user interface. The data is presented in the View. The Dialogue structure contains action buttons and menu's. Buttons are used for the most important functions, whereas other functions are kept in menus which are only accessible in the Teacher mode.

The Prototype

The user interface of the current version of the Intonation Meter, running under MS-DOS, has been developed from a developers point of view. The user interface and the presentation of information were primarily meant for experimental purposes and to facilitate the further development of the system. No special attention was paid to the usability of it in classroom situations or for students practising independently.

The Intonation Meter is currently being transferred to the MS-Windows environment. MS-Windows is becoming a standard environment for interactive applications for MS-DOS machines, especially in educational settings. Programming in MS-Windows facilitates the development of a user-friendly interface with the use of graphics. The prototype version presented here intends to meet the objectives described above. In Figure 5 a possible implementation of the Student mode interface is presented.

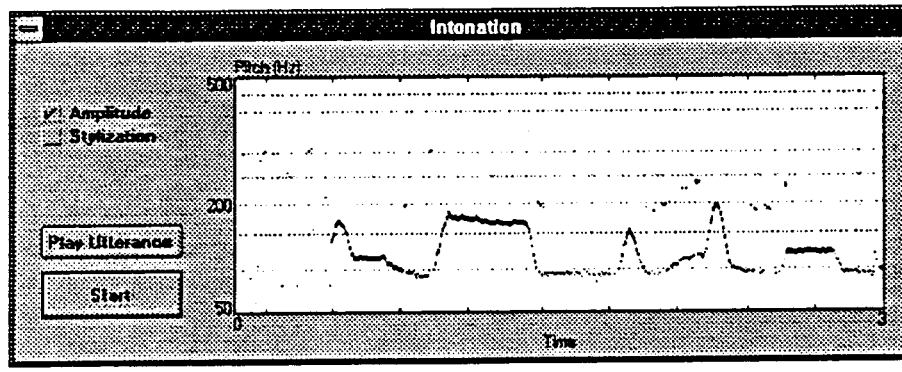


Figure 5. A possible implementation of the interface in the Student mode. The pitch contour and the amplitude of an utterance are displayed. The most important functions, starting and stopping the measurement and playing back the utterance, are accessible here. Some functions and features, like saving and loading, are only shown in the Teacher mode.

When operating in the Student mode, a limited number of functions is accessible to avoid confusion. It should be emphasized that the user interface is still in a stage of development. The design presented here may change in the near future based on further evaluations.

The prototype will be tested in regular speech training sessions. During these sessions feedback from the pupils and the teacher will lead to modifications of the prototype. For the new prototype the same procedure will then be followed. The input of the users of the system will help to modify and improve the usability of the Intonation Meter which may contribute to its use in speech training by making it possible to adapt the system to a variety of teaching techniques and conditions encountered in practice.

Acknowledgments

The work presented here was supported by the Institute for the Deaf/IvD at Sint-Michielsgestel, The Netherlands. The authors would like to express their gratitude to the principals, staff and students of the School voor VSO.

References

Aberton, E., Parker, A. & Fourcin, A.J. (1978) Speech improvement in deaf adults using laryngograph displays. Speech and Hearing. Work in Progress UCL, 33-60.

Angelocci, A., Kopp, G. & Holbrook, A. (1964) The vowel formants of deaf and normal hearing eleven to fourteen-year-old boys. J. Speech Hear. Disord. 29, 156-170.

Bush, M. (1981) Vowel articulation and laryngeal control in speech of the deaf. Ph.D. dissertation, Massachusetts Institute of Technology.

Collier, R. (1975) Physiological correlates of intonation patterns. J. Acoust. Soc. Am. 58, 249-255.

Dagenais, P.A. & Critz-Crosby, P. (1992) Comparing tongue positioning by normal-hearing and hearing-impaired children during vowel production. J. Speech Hear. Res. 35, 35-44.

Friedman, M. (1985) Remediation of intonation contours of hearing-impaired students. J. Commun. Disord. 18, 259-272.

Gold, T. (1980). Speech production in hearing impaired children. J. Commun. Disord. 13, 397-418.

Hudgins, C.V. (1946) Speech breathing and speech intelligibility. Volta Rev. 48, 642-644.

Hudgins, C.V. & Numbers, F.C. (1942) An investigation of the intelligibility of the speech of the deaf. Genet. Psychol. Monogr. 25, 289-342.

Ladefoged, P. (1967) Three areas of experimental phonetics. London: University Press.

Levitt, H., Youdelman, K. & Head, J. (1990) Fundamental Speech Skills Test (FSST). Colorado: Resource Point Incorporation.

McGarr, N. & Osberger, M. (1978) Pitch deviancy and the intelligibility of deaf children's speech. J. Commun. Disord. 11, 237-247.

McGarr, N.S., Youdelman, K. & Head, J. (1989) Remediation of phonation problems in

hearing-impaired children: Speech training and sensory aids. Volta Rev. 91, 7-17.
McGarr, N.S., Youdelman, K. & Head, J. (1992) Guidebook for voice pitch remediation in hearing-impaired speakers. Colorado: Resource Point Incorporation.
McGarr, N.S., Head, J., Friedman, M., Behrman, A.M. & Youdelman, K. (1986) The use of visual and tactile sensory aids in speech production training: A preliminary report. J. Rehab. Res. Develop. 23, 101-110.
Metz, D.E., Schiavetti, N., Samar, V.J. & Sitler, R.W. (1990) Acoustic dimensions of hearing-impaired speakers' intelligibility: segmental and suprasegmental characteristics. J. Speech Hear. Res. 33, 467-488.
Nickerson, R. S. (1975) Characteristics of the speech of deaf persons. Volta Rev. 77, 342-362.
Nickerson, R.S. & Stevens, K.N. (1973) Teaching speech to the deaf: Can a computer help. IEEE Trans. Audio Electroacoust. AU-21, 445-455.
Spaai, G.W.G., Storm, A. & Hermes, D.J. (1993) A visual display system for the teaching of intonation to deaf persons: Some preliminary findings. J. Microcomp. Appl., in press.
Stathopoulos, E.T., Duchan, J.F., Sonnenmeier, R.M. & Bruce, N.V. (1986) Intonation and pausing in deaf speech. Folia Phoniatr. 38, 1-12.
Tye-Murray, N. (1992) Articulatory organizational strategies and the roles of audition. Volta Rev. 94, 243-259.
Van Uden, A.M.J. (1977) A world of language for deaf children. Lisse: Swets & Zeitlinger.
Youdelman, K., MacEachron, M. & Behrman, A.M. (1988) Visual and tactile sensory aids: Integration into an on-going speech training program. Volta Rev. 90, 197-207.
Youdelman, K., MacEachron, M. & McGarr, N. (1989) Using visual and tactile sensory aids to remediate monotone voice in hearing impaired speakers. Volta Rev. 91, 197-207.

Animated icons promote learning of their functions

ALP TIRITOGLU
Department of Design, University of Kansas, Lawrence KS 66045, USA

JAMES F. JUOLA
Department of Psychology, University of Kansas, Lawrence KS 66045, USA

Abstract. In the present research the benefits of animated icons were examined. A subset of static icons were taken from a CAD program and converted into animated icons. Subjects were tested about their understanding of the functions of the icons before and after the animation took place. In addition, a questionnaire was given to find out their subjective opinions about their experience. It was found that animated icons have a positive effect on comprehension of icons and the tasks they represent. In addition, some major issues relevant for effective use of animated icons in user interface design are discussed.

1. Introduction

User interface design is one area in which art and science work together cooperatively to improve the communication between technology and humans. Design principles help designers to some extent, but these design principles come from experience within specific domains of research and applications. If a designer is dealing with a new domain, which is a common case in this era, in addition to existing guidelines, the designer has to use his/her intuition to solve new user interface problems that have never been investigated.

Computer Aided Design [CAD] has undergone incredible growth and variegation in the last twenty years. By bringing new concepts, tools, and methods to drafting and rendering that could not even have been imagined before, CAD systems have changed the way people work on drafting boards. In conventional drawing, designers' tools are limited (e.g., templates, protractor, rules, compass), and it is almost impossible to talk about abstract tools such as zooming, rotation, or extrusion. The tools given by CAD systems are numerous and constantly increasing in new releases. However, adding new tools does not necessarily help users. Frequently, it is not clear how these tools could facilitate users' work. Some tools appear to be so complex that it seems easier to use the old-fashioned way. Although basic functions are the same in most CAD systems, there are

considerable differences among them. These differences are not really in how they accomplish the task, but in their user interfaces. Unfortunately, there is not any standard yet. Early CAD systems did not use Graphical User Interfaces [GUI], instead they carried existing user interfaces into their new releases. Today GUI systems are a large part of CAD systems, and iconic user interfaces are extensively used in GUI systems.

Icons are important elements in graphics-based user-interface design. However, they can be misleading to users if they are not carefully designed. In many cases icons are designed so poorly that they do not make sense, or their meanings are ambiguous. Rogers (1989) stated that the ambiguity of an icon's meaning can be narrowed by the context in which it is displayed. However, this is not enough to prevent ambiguity of icons, particularly for those who have never experienced the system. "One of the main problem with iconic interfacing is that while on some occasions it is relatively easy to interpret the intended meaning of an icon, for others a whole range of different meanings can be attributed to a single icon - each being as well as the other" (Rogers, 1989, p.106). One example from a CAD application is an icon which displays a little mountain. First-time users are generally confused by this icon, thinking that it could be a "perspective view," "overall scene," or some sort of zoom function. The icon becomes more understandable to the user only if the user activates the icon or refers to a help source. Animated icons are alternative solutions to the problem.

Fig. 1. A zoom-in icon in MicroStation ver. 3.5

Animation gives another dimension to icons that may prevent ambiguity and make icons more meaningful. Animation might also help users to understand and follow the steps needed to accomplish a task.

2. Purpose

There are two beliefs about iconic user interfaces. One group in the Human-Computer Interaction [HCI] Community finds icons an efficient means of communication, whereas the other considers them vague and imprecise (Rogers, 1989). The same people who believe that icons are useful think that icons reduce system complexity and are easier to learn, since icons work visually.

Fig. 2. A poorly designed trash box icon can cause confusion.

Rogers (1989) stated that the way in which icons work at the visual level is as pictorial representations of various aspects of an interface metaphor; e.g., a trash can to represent erasing a file. "Icons take advantage of peoples' skills, such as visual recognition and detection of continuity" (Fairchild, Meredith, and Wexelblat, 1989, 132). Icons are also considered as language-independent communication tools. They carry the same meaning for many cultures and nations; e.g., a "trash can" is always a "trash can" even though the shape of it

might vary. However, icons can be ambiguous and deluding. If a "trash can" icon is not designed well (e.g., see Fig. 2), some users may interpret it as a save box which makes their files write-protected or not accessible for other users. At this point icons do not offer any real advantage over other coding methods. Rogers (1989, p.106) points out that "unlike verbal language that is built by a set of syntactic and semantic rules which provide us with a means of disambiguating the meaning of verbal language, in icons there is no equivalent set of rules underlying its comprehension."

In the present research project the potential benefit of *animated icons* is investigated. An existing CAD application (MicroStation[1] 3.5., Intergraph) is taken as a platform. MicroStation is a 2D & 3D CAD application package, and its user interface is based on Graphical User Interface techniques (Direct Manipulation, Windows, and Icons). In MicroStation's user interface, almost every function is attached to an icon. Although some icons are obvious in function, others are abstract on different levels. Twenty-four icons from different levels were converted into animated icons, and subjects were tested before and after animations took place. Since interactive and dynamic programs require constant manipulation, there is a potential for using animated icons in applications for all types of users to increase their learning and comprehension. Do animated icons actually aid people in understanding their functions and accomplishing the task? The present research answers this question more precisely by defining some of the variables in human-computer interaction. It also emphasizes principles and possible pitfalls in designing animated icons for user interface design.

3. Method

3.1 Task Analysis

In the present research, the purpose of task analysis was to understand how users interact with icons on the MicroStation platform. Ultimately, the goal was to create the most useful animated icons for the users. Users' actions in activating icons and how they use the related functions were iteratively examined. According to Galambos and Abelson (1986) people recognize and use structures of events (e.g., orders, sequences of actions, and priorities of events) to understand, explain, and make predictions. Sequences of actions; e.g., drawing a circle, were recorded and used in generating animations. People who were experienced users of MicroStation were interviewed for possible modifications to fine-tune the animations. At different levels of the design process, people from various disciplines were questioned to find out whether the animations made sense to them.

In MicroStation, a mouse or a tablet is a primary input device. Users select the icons by pointing the cursor to the icon and pressing the mouse button. The selected icon becomes an active function waiting for necessary parameters from

[1] MicroStation is a trademark of Integraph, Inc.

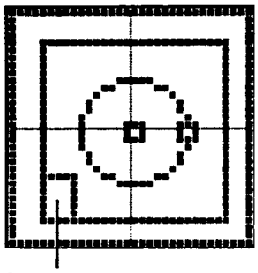


Fig. 3. A 32 by 32 grid in which the icon is defined

the user, such as the center and circumference of a circle. The function remains active until another function is selected.

In the development of animated icons, animations were integrated into the existing static icons. The way users select the icons and how they interact with them was kept the same. However, a new button was added to the icons, at their bottom-left corner, as an animation button. Animations took place in the area in which the icon was defined (32 by 32 pixels) when the animation buttons were selected. Once the animation was completed, the icon turned back to its original shape. Animations could be viewed as many times as the user needed to do so. This gave the user a chance to watch the animations more than one time to complete missing points in his/her observation.

3.2 Simulation of Animated Icons in MicroStation Platform

The MicroStation user interface was designed on the basis of direct manipulation techniques, and almost every graphical function is attached to an icon. MicroStation's user interface was simulated by using the HyperCard[2] background feature with active buttons in the foreground. Twenty-four icons were simulated and converted to animated icons. These icons (functions) were classified into three groups.

1- Non-Abstract, Pragmatic Functions
(i.e., Place a Circle by Center, Modify Element, Delete Part of Element)
2- Semi-Abstract Functions
(i.e., Copy Element, Rotate Element by Active Angle)
3- Abstract Functions
(i.e., Zoom In, Zoom Out from Selected Point)

Except for the selected and implemented icons, all other icons were shaded to keep the same global look and feel of MicroStation, and to make the available icons more distinctive.

3.3 Experiment and Questionnaire

The experiment used 60 subjects who were randomly divided into two equal-sized Control and Experimental groups. Although most subjects were students from various departments at The University of Kansas, some subjects were professors, academic personnel, and community members. Each group participated in a two-part experiment. In the first part of the experiment, subjects were asked to look at the 24 static icons on the screen and write down their possible meanings. At this

[2] HyperCard is a trademark of Apple Computer, Inc.

point the subjects of both groups were allowed to look at the screen and individual icons, but they were not allowed to activate any of the icons or related functions. In the second part of the experiment, the control group was allowed to activate static icons and their functions, whereas the experimental group was allowed to watch animated icons at any time. Control and experimental groups were given equal opportunities to try the functions on the MicroStation platform, except the control group could not activate the animations. The first part of the experiment was limited to 40 minutes and the second part to 75 minutes.

Subjects were asked to complete a three-part questionnaire: prior, during, and following the experiment. Questions asked prior to the experiment primarily related to the subjects' level of knowledge pertaining to computers and icons. During the experiment, and after activating the icons and their functions, control and experimental groups were again asked to describe in writing the meaning of the 24 icons and how they appeared to function during the application. In addition, their verbal comments were noted. Following the experiment both control and experimental groups were asked to complete the third part of the questionnaire, a semantic differential analysis (image profiling) test. The experimental group was also asked to complete an additional questionnaire to evaluate animated icons based on their experience during the experiment.

Each response of subjects for functions of icons was graded between '0' and '4' with '0' the lowest and '4' the highest grade. Evaluation of answers for the icons took place in five categories: 0: No relation or no explanation at all, 1: There are some implications, but no explanation or wrong explanation, and no methods, 2: The function is understood, but the method is not explained or the method is wrong, 3: The function is understood, but the method is not totally correct, 4: The function is completely understood including the steps needed to be taken to operate it. Evaluation of responses and grading were done by the first author and a faculty member in the computer science department at the University of Kansas, who has experience in MicroStation. The grading method was discussed to insure consistency and reliability and was applied equally to the first and second parts of the written test.

4.0 Results

On the basis of the initial questionnaire (see Table 1), subjects within each group were divided into non-experienced, beginner, intermediate, and advanced computer users with Macintosh or other icon-based system experience. For later analysis, the non-experienced and beginner subgroups were combined.

The main performance measures were taken the written descriptions given for each of the 24 icons by subjects before and after working with the system. The results are shown in Table 2, in which each subject's written descriptions which were scored and totaled to produced a maximum score of 96 (24 icons x 4 possible points). The actual data presented are the raw scores converted to percents. Significant differences were found between the improvement scores for the two

Table 1. Answers of both subject groups to the initial questionnaire.

1- How much computer experience do you have?

	None	Beginner	Intermediate	Advanced
Control	**1**	**9**	**13**	**7**
Experimental	**1**	**8**	**16**	**5**

2- How much experience have you had with Macintosh Computers?

	None	Beginner	Intermediate	Advanced
Control	**6**	**11**	**12**	**1**
Experimental	**2**	**7**	**16**	**5**

3- Have you ever experienced Iconic User Interfaces with any computer system?

	Yes	No
Control	**15**	**15**
Experimental	**22**	**8**

4- If your answer is "Yes" for the previous question:"
Do you think Iconic User Interfaces are fun to play with?

	Yes	No	Sometimes	(Blank)
Control	**9**	**-**	**6**	**15**
Experimental	**12**	**2**	**10**	**6**

5- Do you think icons usually carry the meaning of what they refer to?

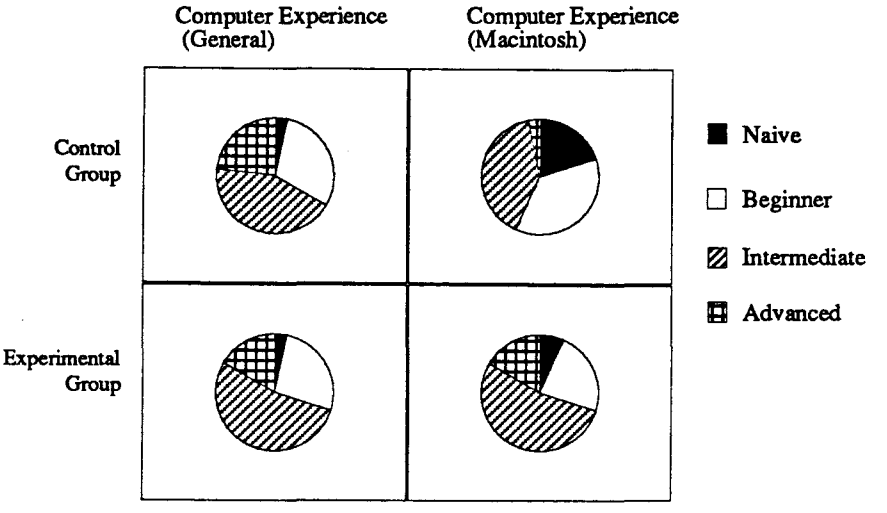
	Yes	No	Sometimes	(Blank)
Control	**12**	**-**	**10**	**8**
Experimental	**7**	**4**	**14**	**5**

6- Do you think icons adequately explain everything you need to know about that specific task?

	Yes	No	Sometimes	(Blank)
Control	**1**	**6**	**15**	**8**
Experimental	**-**	**12**	**13**	**5**

7- Have you ever seen animated Icons in an application?

	Yes	No	(Blank)
Control	**2**	**22**	**6**
Experimental	**3**	**24**	**3**



groups. Specifically, the experimental group showed a greater gain in accuracy of described function for the icons overall and for the beginning and intermediate experience groups. There was no significant difference in the improvement scores for the advanced experiental and control groups. This result was probably due to a subject selection error in that the three best subjects in terms of performance on Part 1 of the test were all in the advanced experimental group, and ceiling effects limited the amount of gain they were able to show in Part 2. Overall, however, the 36% gain shown by the experimental group is highly significantly greater than the 21% gain shown by the control group.

Figure 4 shows the semantic differential mean scores for the two groups of subjects, which clearly indicate a higher overall impression of the system for the experimental group. These results are supported by the indication of the experimental group expressed in the final questionnaire (Table 4) that they would generally like to have animated icons available in other applications and as help to indicate icons' functions.

Table 2. Mean rated scores (percent) for each subject in the Control and Experimental Groups. Part 1 data were collected prior to working with the system and Part 2 data were collected afterwards. The heavy lines seperate beginners, intermediate, and advanced groups who are compared using t-tests.

Control Group			
Subjects	Part 1	Part 2	Difference
1	8.3	32.3	24.0
6	19.8	21.9	2.1
11	21.9	33.3	11.5
14	10.4	35.4	25.0
16	7.3	35.4	28.1
17	29.2	65.6	36.5
23	25.0	41.7	16.7
24	14.6	19.8	5.2
26	19.8	38.5	18.8
28	65.6	68.8	3.1
3	40.6	44.8	4.2
5	15.6	30.2	14.6
8	29.2	49.0	19.8
9	39.6	74.0	34.4
10	24.0	46.9	22.9
12	44.8	53.1	8.3
15	1.0	10.4	9.4
18	19.8	24.0	4.2
19	3.1	20.8	17.7
21	47.9	59.4	11.5
22	22.9	36.5	13.5
25	3.1	43.8	40.6
27	25.0	58.3	33.3
2	30.2	30.2	0.0
13	26.0	91.7	65.6
4	76.0	67.7	-8.3
7	21.9	78.1	56.3
20	3.1	49.0	45.8
29	4.2	49.0	44.8
30	34.4	66.7	32.3
Mean	24.5	45.9	21.4

Experimental Group			
Subjects	Part 1	Part 2	Difference
2	11.5	52.1	40.6
4	27.1	77.1	50.0
5	21.9	57.3	35.4
6	4.2	68.8	64.6
8	25.0	57.3	32.3
14	16.7	78.1	61.5
17	40.6	87.5	46.9
19	14.6	34.4	19.8
23	24.0	77.1	53.1
1	20.8	39.6	18.8
3	51.0	85.4	34.4
9	79.2	92.7	13.5
10	31.3	71.9	40.6
11	11.5	74.0	62.5
13	14.6	75.0	60.4
15	16.7	60.4	43.8
16	44.8	78.1	33.3
18	24.0	60.4	36.5
20	18.8	58.3	39.6
21	21.9	67.7	45.8
22	22.9	77.1	54.2
25	50.0	70.8	20.8
26	39.6	71.9	32.3
27	41.7	81.3	39.6
28	60.4	67.7	7.3
7	67.7	83.3	15.6
12	65.6	84.4	18.8
24	19.8	63.5	43.8
29	96.9	95.8	-1.0
30	68.8	86.5	17.7
Mean	35.1	71.2	36.1

t-Test: Two-Sample		
Beginners	Control	Experimental
Mean	17.08	44.91
Variance	133.65	207.10
Observations	10	9
df	17	
t	-4.67	
p	<0.0003	

Intermediate	Control	Experimental
Mean	18.03	36.46
Variance	138.94	247.28
Observations	13	16
df	27	
t	-3.50	
p	<0.002	

Advanced	Control	Experimental
Mean	33.78	18.96
Variance	783.72	256.81
Observations	7	5
df	9.70	
t	1.06	
p	=0.315	

All Groups	Control	Experimental
Mean	21.39	36.08
Variance	309.63	295.26
Observations	30	30
df	58	
t	-3.27	
p	<0.002	

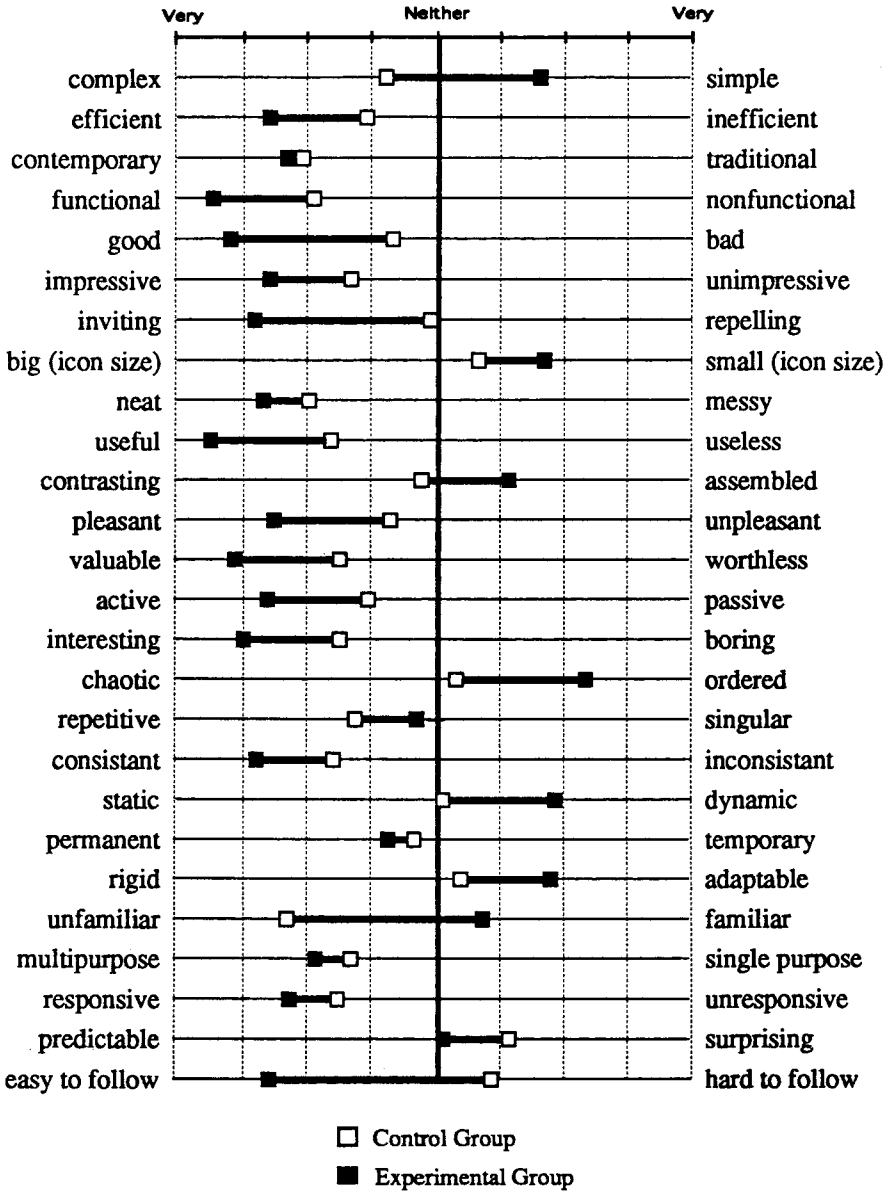


Fig. 4. Control and Experimental Groups' Semantic Differential Analysis

Table 3. Answers for the final questionnaire (Experimental Group only)

1- In general, do you think animations explain better what Icons stand for?

Yes	No	Sometimes
29	-	1

2- Do you think Animated Icons are too fast to follow?

Yes	No	Sometimes
-	23	7

3- After seeing the animation, do you still feel like you need to get into the icon and try to do something?

Yes	No	Sometimes
6	3	21

4- Do you feel like you need to watch the same Animation over and over to understand what it means?

Yes	No	Sometimes
-	9	21

5- After seeing Animated Icons, would you look for the same feature in another application to assist you, or would you still prefer static icons?

I may look for the animation feature	**28**
I prefer static Icons	**-**
I don't know	**2**

6- When you see animated Icons, do you feel that the application that you see is not a sophisticated and serious application?

Yes	No	I don't care
2	25	3

7- Do you find it difficult to animate Icons?

Yes	No	Sometimes
2	18	10

8- Do you think after you learn the system you might still need some of the Icons remaining in the animation mode?

Yes	No
20	10

9- If you have to grade the Animations you have seen what would be your personal opinion? (0: the lowest, 10:the highest)

AVERAGE: 8.65

5.0 Discussion

The results showed that the experimental group, who had experience with animated icons, improved 15% more in acquisition of knowledge about icons' functions than did the control group, who had experience with static icons, during a short session in which they used the icons in a CAD environment. It needs to be mentioned that all of the animations developed in this research were based on existing static icons. In other words, the starting point in designing animated icons was not a set of novel animated icons, but an existing, established set of static icons. It is presumable that if the same icons had been designed as animated icons from the beginning, the increase in understanding of their meanings would be even more substantial than that shown here.

Some subjects mentioned having difficulties in finding the right icon to select. It appears to be that dealing with an iconic user interface, in which icons are used extensively, can be overwhelming and actually makes the task difficult for novice users. Trial-and-error definitely helps the users to understand the functions. Trial-and-error in learning can be prevented to some extent with animated icons, but may not be completely eliminated.

Association between the symbol in the icons and the objects they refer to is sometimes indirect. For example, in the second part of the experiment some subjects were confused with the shape given in "move" or "copy" icons. These icons show rectangular shapes, however their functions are capable of moving or copying any entities on the screen. Therefore, novice users think that that specific icon can work only with rectangular shaped objects. This is a good example of poor mapping, and it should be prevented from the very first at the design stage of the system. We can see the same problem with "Fence" functions in which the iconic representations of the functions make sense only after those functions are learned. The intention of the icons should be the other way around, and the user should be able to learn what icons stand for prior to the usage of the functions.

Some icons require operations or objects prior to themselves (e.g. move, copy, or fence operations). In other words, the user is supposed to use the functions to change the state of an existing object. Such icons usually do not carry this information at all. Including animation can be a solution to this problem, to indicate that prior operations or objects are required for specific icons. Another way to deal with this problem is to turn off those functions that cannot be used (e.g., dimmed icons, or faded icons).

In terms of subjective opinion, from the post experiment questionnaire for the experimental group, we can say that almost all of the subjects (29 of 30) think that animated icons better explain what the icons stand for than do static icons . However, most of these users also agree that they still need to try some of the functions to absorb them. Most subjects think that animated icons can be used as reminders during their use of applications.

Help features, such as animated icons, do not necessarily improve understanding of the system for advanced users. However animated icons could be used as

reminders if the applications have many functions to remember, such as in MicroStation.

The semantic differential analysis gives us a compact result about subjects' personal opinions. From the profile (see Figure 4) we can see that subjects think animated icons feel simpler to use than static icons. Overall, clearly animated icons have more positive effect on users than static icons.

5.1 Cognitive Aspects

Interaction between the system and the user occurs in two ways; *system to user communication* and *user to system communication.* A CAD system generally provides four types of information: *graphical information*; e.g., the representation of CAD database on the screen, the design object; *diagnostic and verifying information;* e.g., numerical data, coordinate values, angles, system status; *operation menus;* e.g., functions and macros; and *prompts;* e.g., guides which help the user perform a sequence of operations, telling what data are needed. Conventionally, user to system communication includes a graphical input device and an alphanumeric keyboard. These devices allow the user to logon/logoff the system, to choose the operations, and to manipulate the image. According to Waern (1988) thirty percent of CAD commands involve choosing operations, such as dimensioning, moving details from one place to another, or turning operations (Semi Abstract Functions). Thirty to fifty percent of the CAD commands are creating and changing operations of the image on the screen (Pragmatic Function), and 20% to 30% of the operations are verification, saving, and checking operations (Abstract and Semi-Abstract Functions).

In CAD, every operation must be achieved with a sequence of keyboard or mouse operations. Therefore, drawing in CAD is not a direct task. For beginners, this is a problem, since they have to spend considerable time learning the user interface of the system as well as abstraction. For example, Waern (1988) pointed out that even though zooming is a very efficient property of CAD systems, it may present short-term memory problems. The user may lose the perception of totality of the design object.

Animated icons should tend to facilitate visualizing complexity and conceiving of tasks and problems in visual terms. "The mind, reaching far beyond the stimuli received by the eyes directly and momentarily, operates with the vast range of imagery available through memory and organizes a total lifetime's experience into a system of visual concepts. The thought mechanisms by which the mind manipulates these concepts operates in the direct perception, but also in the interaction between direct perception and stored experience, as well as in the imagination of the artist, the scientist, and indeed any person handling problems 'in his head'" (Arnheim, p. 38, 1969). Visualization is an important concept in user interface design. Visualizing what is happening facilitates learning and helps in constructing a mental model of the system.

In a CAD application there is usually more than one way to accomplish a

specific task, but the question is to find a way to perform the task efficiently and in the shortest amount of time. Animated icons help to use a function properly as well as to learn the system with minimal trial and error. Ambiguity is a problem with icons in terms of the image presented. When images are simple, there is a possibility of ambiguity that may lead users to misinterpret the icon. Hanson (1965) indicated that the expert and novice see things differently due to their expectations and knowledge. Wertheimer (1959) pointed out that the same problem can be solved more easily by restructuring of a figure, sometimes by including motion. Motion does not bring a third dimension, however it helps to visualize how things are related to each other and how they interact with each other successively in space and over time.

Many problems can be solved by manipulating images. Some recent studies indicate that the ability to manipulate spatial information can make human-computer interaction tasks easier to perform (Eberts, 1989). Spatial manipulation may be useful in solving problems as well as in teaching tasks. Eberts found that showing how tasks are performed with computer graphics was beneficial when the subjects were later asked to solve problems when the graphics were not available. Graphics and spatial reasoning aided through animation facilitate how users think of the task and facilitate the users' internalization of the information.

Memory is an important factor in the design process, because it is a critical limiting factor of human information processing. In a CAD application such as MicroStation, the number of functions and how they are performed are almost impossible to remember unless the user has years of experience. Animation is a quick way to remember and refresh the memory through visualization. In general, recall is better if the context of remembering fits the learning context of memorizations. In that sense, it seems to be that in a graphical working environment, graphic-based help works better than text-based help, since the user does not have to convert the text information into an image to visualize it.

Rogers (1989) studied the effectiveness of using icons as opposed to command names. In his experiment, subjects' recall of the meaning of icon-mapping improved over time in a similar way for both the icon set and the command name set. However, when they were later asked to recall what the names meant, subjects in the icon condition outperformed those in the command name condition.

Consistency is another important factor in learning. We know that it is easier to recognize patterns in consistent environments. Regardless of where the animations are used, in animated icons or in animated help screens, they should be consistent in terms of presenting common clues. The user could then determine some likely features when they see new animations. It appears that learning is mostly under the control of the subject. However, the way information is organized is a key factor in learning. Information organization is usually based on order, classification, or semantic relations among entities. Animated icons facilitate the learning process by giving semantically organized information about the functions and the syntax of their use. In addition, animations aid in seeing the whole picture of the concept presented with less cognitive load than many other forms of communication media.

Abstract icons require more time to learn than pragmatic icons, such as rotating an entity vs. drawing a rectangle. It seems that users have a better understanding and learning of activities associated with relevant past experience. In that sense, for a designer who has experience on a drafting board, icons, which are similar to drawing templates, take less time to learn. However, abstract icons and related functions seem to be equally difficult for both experienced and novice users. During the first part of the experiment, it was observed that regardless of their experience level, all subjects had the same difficulties in understanding certain abstract icons; e.g., fence manipulations. After viewing the animation, there was a significant increase in comprehension of abstract icons. However, animation does not change the fact that abstract functions require more trial and error in order to be understood and learned completely as opposed to empirical functions.

It is expected that people who have acquired extensive knowledge and skill related to a job could perform better in using computer systems on the job than users with little domain specific knowledge. However, this is not always the case. Domain specific knowledge does not necessarily predict success in interaction with a computer system. Domain specific knowledge shows its effects only after users have acquired some experience with the computer interface (Egan, 1988). Animations and animated icons do not necessarily teach domain specific knowledge, but they can simplify the learning of the interface by showing the actions. Consequently, the user spends less time to develop new skills in order to understand the application.

Animated icons appear to be helpful in constructing a mental model of the system for the users. However, animated icons can also be misleading due to the fact that they are usually simplified to be more understandable. Therefore, in designing animated icons, the designer should consider the fact that the scenario chosen for the animation has a direct impact on users' mental models of the system.

5.2 The Design of Static and Animated Icons

5.2.1 Issues in Designing Static Icons

There is not any standard procedure to create icons in user interface design. However, there are some major issues that the designer should keep in mind to create successful and meaningful icons. Some of these issues are also valid for animated icons.

Static icons should be designed as clearly as possible so that the user does not have to learn the meaning of the icon through another sources such as on-line help or manuals. Although establishing a direct map between the real world and the icon is difficult to obtain, it plays an important role in the success of icons. Symbols are also used in icon design, but they require an initial learning procedure. Sutcliffe (1989) stated that an absolute boundary between symbols and

icons is illusory because as soon as a symbol's meaning has been learned it will become a meaningful image. On the other hand, an icon may be ambiguous or have no immediate meaning even though it is a complex and apparently realistic image.

The size of icons is another important issue in designing effective icons. Icons are usually based on a sixteen, thirty-two, or sixty-four pixel grid system depending on the resolution of the system and the details of the icon. According to Sutcliffe (1989) simple icons can be effective in dimensions of 0.5 cm. square and complex icons can be successful in 1.0 cm. square. Sutcliffe (1989) also noted that giving a clear outline is also important to help visual discrimination.

In any system icons should carry consistent graphical characteristics such as line thickness, patterns, and colors. In addition, graphical elements that distinguish an icon from other icons must be clear to the user (Brown, 1988).

Most of these issues can also be applied for designing animated icons. However, animated icons introduce some other issues such as scenario development, visualization and more important, motion. Animated icons may ease some of the abstract static icon designs by replacing them, but they definitely open new questions that demand research in the field.

5.2.2 Definition of Animation

Baecker and Small (1990) define animation as follows: "Animation is not the art of DRAWINGS-that-move but the art of MOVEMENTS-that-are-drawn. What happens between each frame is more important than what exists on each frame. Animation is therefore the art of manipulating the invisible interstices that lie between frames. The interstices are the bones, flesh, and blood of the movie, what is on each frame, not merely the clothing." (Baecker and Small, p. 251,1990).

There are a number of reasons to use animation at the interface. Baecker and Small (1990) stated some of these as follows:

- to visualize and simulate a class of functions
- to orient the user during transitions from one process to another
- to identify an application quickly and vividly when it is invoked
- to provide an overview of complex menus
- to improve the information content of icons or symbols used to perform actions
- to provide more interesting, understandable, and communicative tutorials and explanations
- to provide current information concerning the status of a system or its background processes
- to provide access to interaction history and historical context
- to improve the delivery of error messages and help and thereby accelerate the correction of mistakes

Developing a scenario for icons is directly related with icon mapping in time. First of all, icons have to be designed as part of the animation, not as a separate

icon that follows an animation. Secondly, besides making functions clear, animation is capable of increasing the information represented and showing the necessary steps to be taken to activate their related functions. "A great deal of trial-and-error may be required to understand what is going on. Animation can clarify underlying relationship and introduce functionality" (Baecker and Small, p.261, 1990). Therefore, task analysis is necessary to extract the points and steps for each function. Animation should be kept in simple form. There are two main reasons for this: first, complex animation requires significant computer power, and second, simple forms of animation can be more effective in terms of learning. The following steps are key factors in developing a scenario in animated icons:

- Analysis of task domain
- Identification of the function
- Analysis of basic characteristic of the function
- Analysis of steps to be taken to execute the function
- Analysis of integration of the function with respect to the system

6.0 Conclusion

6.1 Research Conclusion

In the present research the usability and effectiveness of animated icons were examined. Although some animated icons have been used in the past, these icons were used only to show the current state of a system (e.g., the level of gas in a gas-tank, the water pressure). The present research examined animated icons in the context of an educational tool, a tutorial, a help utility, or a reminder for complex functions. Overall, a fifteen percent increase has been observed in understanding of functions by subjects with animated icons as opposed to static icons. The results indicate that there is a significant potential in using animated icons in user interface design, not only for operating system level user interfaces or CAD applications but also for any product that may use animated icons (e.g., audio systems, VCRs, Copy Machines, and Cameras).

From the results, we have seen that animated icons have greater advantage over static icons for abstract operations. Especially, abstract functions, such as zoom, can be designed more explicitly by using animated icons.

Motion picture and current animation techniques may help to understand some issues in designing animated icons, but further research is necessary to understand cognitive issues of animated icons.

It should be noted that all the animated icons were presented in a CAD environment. Therefore the results obtained from this research generalize primarily to how users interact with animated icons in an environment in which domain specific knowledge is required. We can assume that if animated icons help users who are not literate about the domain, they would definitely help those who have domain-specific knowledge. This assumption suggests that animated icons could be used as a tutoring tool to teach syntax of an application for advanced users.

6.2 Future Research

Animation is an effective means for instructional purposes. It can be used in many ways in user interface design. For instance, instead of showing the animation in the limited area of the icon, animations can be shown in a window which is opened from the icon itself. Since more space is gained by opening a new window, more elaborate animations can be presented. However, there is no guarantee of the success of such features without further research.

In general, icons are two-dimensional pictures. In recent user interface systems, a pseudo third dimension has been used in many icons (i.e., usually a shading around the icon, e.g., buttons, switches). The pseudo third-dimension mostly helps to make representational icons more realistic. However, a real three-dimensional icon has not yet been created. We have no idea what kind of results we can expect if we create an environment that uses three-dimensional icons, such as solid icons for user-computer interaction. It seems that virtual reality is one field in which three-dimensional icons can be used experimentally. Tools and functions would become more realistic than their two-dimensional representations. In this case, perhaps the users would hold the icons, change their parameters (object-oriented icons; e.g., an eraser icon would erase part of a raster image but also the same eraser icon would delete an entity if it is used in vector-based image), and use them for their needs.

If we include a third dimension and motion into icons, then we should question the term "icon." It this case, we may leave icons in their two-dimensional environment and call our new tools "*idols*".

References

Arnheim, Rudolf. (1969) "The Intelligence of Visual Perception" *Visual Thinking* Univ. of California Press., 1969. 37-53.

Baecker, Ronald., Small, Ian. (1990) "Animation at the Interface" *The Art of Human-Computer Interface Design* ed. by Brenda Laurel Addison-Wesley Publishing Comp., Inc., 1990. 251-267.

Brown, C. Marlin "Lin". (1988) *Human-Computer Interface Design Guidelines.*, Ablex Publishing Corp. 1988.

Eberts, Ray E., Eberts Cindelyn G., ed. by Hancock, P.A., Chignell. (1989) "Four Approaches to Human-Computer Interaction" *Intelligent Interfaces*. Elsevier Science Pub., 1989. 69-127.

Egan, Dennis E. (1988) "Individual Differences In Human-Computer Interaction" *Handbook of Human-Computer Interaction* 1988. 543-565

Fairchild, Kim., Meredith, Greg., and Wexelblat, A., (1989) "A formal structure for automatic icons." *Interacting with Computers* Butterworth & Co (Publishers) Ltd. 1989. vol 1 no 2: 131-140.

Galambos, J. A., and Abelson, R.P. (1986) "Knowledge Structures for Common Activities" *Knowledge Structures*, 1986.
Hanson, N. R. (1965) *Patterns of Discovery* Cambridge Univ. Press, 1965.
Lodding K. N. (1983) "Iconic Interfacing" *IEEE Computer Graphics and Applications*, March/April 1983.
Rogers, Yvonne. (1989) "Icons at the Interface: their usefulness." *Interacting with Computers*. Butterworth & Co (Publishers) Ltd. 1989. vol 1 no 1: 105-117.
Sutcliffe, Alistair. (1989) "User Psychology" *Human-Computer Interface Design*. Springer-Verlag Inc., 1989. 7-48.
Waern Karl-Gustaf. (1988) "Cognitive Aspects of Computer Aided Design" *Handbook of Human-Computer Interaction* Elsevier Science Publ., N. Holland., 1988. 701-708.
Wertheimer, Max. (1959) *Productive Thinking* New York, Harper, 1959.

Generating Natural Language in an Immersive Language Learning System

Henry Hamburger[1], Dan Tufis[2] and Raza Hashim[1]

[1] Dept. of Computer Science, George Mason University, Fairfax, VA 22030 USA

[2] Center for Research on Machine Learning and Language Processing
Romanian Academy, Calea Victoriei 125, 71102, Bucharest 1, Romania

Abstract: Our computer-based conversational partner is designed to provide practice for beginning foreign language students. The conversations take place in realistic situations that are expressed both visually and verbally, so that the student can see what he/she is talking about. Besides clarifying meaning, this situational immersion also insures contextual appropriateness. Natural language generation plays a crucial role in such a system, especially for interacting with a beginner, since clearly the learner must first be exposed to the words and constructions of the new language before being expected to produce them. In this paper we therefore stress generation, especially its so-called strategic aspects, in relation to the situations in which the student is immersed.

Keywords: ICALL, language immersion, knowledge representation, interlingua, natural language generation

1 Overview: Language tutoring and language generation

This paper describes the techniques and crucial role of natural language generation - in particular its strategic component - in a foreign language learning and tutoring system that is based on conversation in meaningful situations. Such a learning environment is motivated by important methodological trends in foreign language pedagogy. The system engages the student in tightly integrated linguistic and graphical two-medium communication. For this reason and because the conversation is grounded in realistic situations represented in coherent, realistically organized microworlds, we say that the system is immersive: the student is immersed in a situation.

By taking a generative approach to dialog management, the current system, FLUENT-2 (Hamburger and Hashim, 1992), dramatically enhances the linguistic range of the tutor, in comparison to the completed FLUENT-1 prototype. Both systems attempt to resolve the tension between educational and conversational continuity delineated in Hamburger and Maney (1991). Briefly put, the two continuities mean that the system must say things that make sense in the situational context yet do not violate the precept that new linguistic material should be introduced gradually, according to a pedagogically sound syllabus

FLUENT-2, implemented in Macintosh Common Lisp, is designed to communicate both graphically and linguistically, and in both directions. However, here we will address only the generation of language. Natural language generation is widely viewed as a multi-level process. Our own system has two major processing levels. The first process specifies the message to be conveyed to the

student and the other actually produces it. These two levels are sometimes called the strategic and tactical levels of generation. Often, as in our case, the two levels use different representations, one oriented to problem-solving and the other to linguistic matters. Therefore a third process is required, to bridge the gap between these two representations. Our emphasis here is on the strategic level.

Strategic generation is accomplished principally by the view processor. This module operates on domain knowledge representations - including plans, actions, objects, their properties, classes and their inheritance hierarchies - to produce a language-independent representation of the output sentence. This resulting frame-style structure, which we call the KR of the sentence, reflects the viewpoint of the tutor, that is, what situational aspect - or view - the tutor has chosen to guide what it says. Many views are implemented by extracting and manipulating structural information from such places as the various slots of an instantiated action rule or its arguments, which are objects in the current microworld. It is up to the tutorial strategy module - or tutor - to select a view that both makes sense in the situation and will yield linguistic output that is comprehensible to the student.

Tactical generation is handled by the generation side of the natural language software of the Athena Language Learning Project or ALLP (Felshin, 1993). The ALLP generator operates on a linguistically oriented semantic representation called interlingua (IL) to produce a sentence, phrase or paragraph in any of several natural languages. The ALLP system maintains some focus information and, if provided with appropriate information, takes care of several discourse phenomena.

2 From action to language

Essential to our approach to carrying on a conversation about a dynamic situation is the ability to reformulate a given microworld entity into something to say. Actions are an especially useful basis for linguistic outputs, so we focus on them. In some cases, the tutor's linguistic move is to comment on the action just carried out in the microworld by either the student or the tutor, leading to a description of the action itself or how it changes something. Alternatively, having chosen a potential action, the tutor may tell the student to do it, or ask the student something about it. Yet another class of linguistic moves is that in which the tutor produces a follow-up reaction to a student response. If, for example, the student's graphical move is unresponsive to the preceding command, the tutor may formulate a statement involving two actions - contrasting the actual to the expected one.

Where two actions are involved, as in the preceding example of an action-based command and the graphical response, it must be possible to compare the meanings of the two. Even though one of the actions shows up outwardly in language while the other has arisen from a graphical move, the comparison must take place within a single representation system, specifically that of the microworld. Comparing the two actions is not merely a matter of an equality test, since the action commanded may be less specific than a suitable response. Moreover, if the tutor has told the student to execute a plan rather than an action, the job of evaluating the responsiveness of the student's next action becomes

significantly more difficult, since many kinds of actions may advance the plan, given that it is not, in general, merely an inflexible linear sequence of actions.

The microworld-oriented representation - the KR - that the view processor builds has two parts, WHAT-TO-SAY and HOW-TO-SAY, corresponding to the distinction between the basic meaning of an instruction, question or comment and the specific way of linguistically realizing it. More specifically, WHAT-TO-SAY includes the objects, events, states, and relationships that are to be linguistically realized, whereas HOW-TO-SAY includes tense, aspect, mood, voice, reference definiteness, and so on.

After the view processor has determined what to say and how to say it, the KRIL module translates from these domain KR structures into IL structures. It is important that this translation produce IL structures that will yield smooth natural language output. For instance, the IL should specify the appropriate use of pronouns for repeated reference to the same object in the same sentence or for anaphoric reference to a previous clause, sentence or even graphic move; e.g., "You can't do *that*."

The mapping from KR to IL relies heavily on information provided by the dictionary of the language in use. From the lexical entries, the KRIL module extracts the information it needs to build IL fragments which are further integrated into the final IL structure. This lexical information, which is especially important in the case of verbs, includes such things as thematic structures, selectional restrictions and prepositional attachment requirements. Since the process of building an ILS is driven by this kind of language-specific lexical data, the resulting ILS is partially language-specific. Nevertheless, it is important to note that the process itself is language-independent at the level of code.

3 Generation criteria at the dialog level

Interaction Types. Merely specifying the action on which it is based does not determine the tutor's linguistic output. We have already noted the variety afforded by views, and there are also several dialog factors that further enrich the choice. Perhaps most obvious of these is the interaction type, which specifies: an assignment of roles in the dialog for the tutor and the student, which one of them initiates the interaction, whether each particular turn consists of action or language, and for each linguistic turn, whether it is a command, question, answer, comment, suggestion or description. Eight simple two-move interaction types arise just from the fact that either the tutor or student can initiate the move sequence with an action, command, question or statement (followed by an appropriate response). Particularly relevant to language generation are Movecaster, in which the student carries out any action and the tutor comments on it, Commander in which the student acts in response to a command, and Quizmaster, in which the tutor poses a question.

The choice of interaction type is one of the crucial decisions by the tutorial strategy module, since it has powerful effects on the level of difficulty for the student. The various interaction types provide important variations in sentence structure by requiring declaratives, imperatives and questions, thereby enhancing language experience and also affecting difficulty. The interaction type also

influences the values chosen for some surface generation parameters, including tense, mood, aspect, voice and person. Interaction type also affects which information-level views are appropriate, thereby indirectly giving the interaction type further impact. Additional dialog factors that are not controlled by the interaction type include whether a request is direct or indirect, formal or informal, specific or generic, intensional or extensional.

Closely related to the interaction types are extensions of them that deal with follow-up language suitable for the learning and tutoring situation. This additional kind of dialog factor consists of what we call "evaluative continuation types," (Hamburger et al., 1993) which let the tutor approve, reject or correct a student's graphic move, or assist a student's linguistic move. More such continuations and other interaction types are under development, as well as work allowing for the possibility of breaks for conversational repair. Finally, note that, by their nature interaction types play a key role in dialog management by making successive turns coherent. At the next level, successive interactions should also cohere. To achieve this, we plan to use dialog schemas expressed as transition networks built using interaction types.

Language Structure. Of particular utility to the tutor for adjusting the level of difficulty are language maneuvers that permit the inclusion or omission of some of the arguments, for example, ergative constructions, so-called truncated passives, causatives ("someone fed the baby" vs. "the baby ate") and the inclusion or omission of various kinds of optional phrases. Depending on the interaction type and on evaluative continuation, a decision can be made on producing contrastive and/or elliptic phrases ("The cup not the pot!").

Given that a particular object is to be referred to, various aspects of language can work to identify it uniquely. If it is the only instance of its type, its class name suffices, e.g., "the cup." This is also the case if it is uniquely prominent in some way, by virtue of mention or action. If the set of objects in the class exceeds one, it may be possible to find one or more properties to use for modification to get a set with only one element or only one prominent element, e.g., "the gray cup" meaning the only one at all or the only one on a tray being offered. Properties can be immutable, as in the case of color, but need not be. If not, they may be current or historical, e.g., "the cup that you used to water the plant". A property may pertain only to the object itself or can relate it to another object, by such relationships as location, possession, possessor and function. Abstract relationships are also possible, including order, comparison and exclusion.

4 Generation criteria at the microworld level

At the microworld (henceforth MW) or domain level, there are several decisions to be made that together we refer to as view selection. Views are abstractions of what the tutor can say. They have to do with an action itself and other things in the MW that are related to it, including objects, plans and possibly time, as well as certain aspects of the dialog. Since there are many views, it will help to organize them into categories. To do so, note that in response to some particular action, the tutor can comment on the action itself, the resulting state, or on a subplan,

say the one that has just been completed or the one that can now commence. Thus there are action views, state views and plan views.

The two principal inputs to the view component are the current view (selected by the tutorial strategy module) and the current event, in the form of a fully instantiated action rule. The relevant slots of the action rule for this purpose are its header, goal and knowledge updates. The output of the view component is the two-part KR described in Section 2.

The most straightforward of the action views is one that simply presents an action without reference to context, moreover choosing the most prominent action. In Movecaster, this is the action that the student has just carried out. In Tourguide or Commander, the action is one chosen by the tutor. Although the choice of action may be based on a plan, the plan is not mentioned in this view. The interaction type determines the tense, mood and aspect. Commander, of course, takes the imperative, while Movecaster uses a (recent) past (e.g., "Elle vient de ramasser la tasse").

Negative views are those that can result in a comment on the non-occurrence of a possible action or make a negative command. The action might be one that was anticipated, perhaps because of being next in a plan, in which case it can (but need not) be part of a more complex plan view described below under plans. Other non-occurring events can be constructed in relation to the action that did occur by using the same operator with another argument. These views seem most appropriate when used in combination with others. An example of combining a negative view with a view for an action that did occur is (in Movecaster) "He did not pick up the spoon. He picked up the cup." or more flowingly, "He picked up the cup, not the spoon." A Commander example that contrasts actions rather than objects is "Do not put down the cup. Use it to get some water."

There are additional categories of views. For example, commenting on an action at various relative times or in relation to the time of surrounding events gives rise to temporal views. It is even possible to have "rejection views," for dealing with failed, discouraged or impermissible actions, to correct misperceptions and repair communication.

Acknowledgement This work was supported by grant IRI-9020711 from the U.S.National Science Foundation.

References

Felshin (1993) A Guide to the Athena Language Learning Project Natural Language Processing System. Copyright, Massachusetts Institute of Technology.

Hamburger, H. and Hashim, R. (1992) Foreign language tutoring and learning environment. In Swartz, M. and Yazdani, M. (Eds.) Intelligent Tutoring Systems for Foreign Language Learning. New York: Springer-Verlag.

Hamburger, H. and Maney, T. (1991) Twofold continuity in language learning. Computer-Assisted Language Learning, 4, 2, 81-92.

Hamburger, H., Tufis, D. and Hashim, R. (1993) Structuring two-medium dialog for learning language and other things. Proc.of the ACL Workshop on Intentionality and Structure in Discourse Relations. Columbus, Ohio, June 20.

IMPROVING FUNCTIONAL PROGRAMMING ENVIRONMENTS FOR EDUCATION

J. Ángel Velázquez-Iturbide

Dpto. de Lenguajes y Sistemas Informáticos e Ingeniería de Software, Facultad de Informática, Universidad Politécnica de Madrid, Campus de Montegancedo s/n, Boadilla del Monte, 28660 Madrid, Spain

Abstract. Nowadays, most functional programming environments are inadequate for educational purposes. The minimum requirements these environments should satisfy to be successful can be summarized as: integrated environments with syntax and semantic based tools. A proposal for an environment which satisfies these requirements is given. Early efforts to develop an environment prototype (named HIPE) along these lines are also described.

Keywords. Functional programming, integrated programming environments, semantic-based tools.

1 Introduction

Functional programming languages have experienced a great advance in the eighties, evolving from Lisp language dialects to modern functional languages [3]. The syntactic and semantic improvements the later languages provide and the simplicity of functional paradigm has promoted functional programming as an increasingly frequent subject matter in computer science education [5].

Surprisingly, programming environments for modern functional languages are usually very crude. Even worse, most implementations are commonly not available for personal computers, a kind of machine very common in software laboratories. This is a consequence of the fact that functional programming is still in an experimental phase. The lack of environments adequate for educational use is an important tie for the expansion of this paradigm. However, the syntactic and semantic simplicity of functional languages allows us to hypothesize that building such environments will not be too costly, compared to the potential profit.

The paper describes the minimum requirements a functional programming environment should fulfil to be successful in an educational use and the author's experience in building a prototype along these lines for the functional programming language Hope$^+$. The paper is structured as follows. Section two describes the requirements we identify for educational functional programming environments; different desirable characteristics and tools are identified. The third section describes the prototype mentioned before, named HIPE. In section four we describe some of the improvements we plan for the prototype and our experience with the environment.

Familiarity of the reader with functional programming is assumed in the paper; the interested reader should refer to [2, 4] for more information about Hope$^+$ language, [5] for a modern course design on functional programming, [9] for details of the implementation of HIPE, and [8] for a detailed description of HIPE, including a comparison with related environments.

2 Requirements for an educational functional programming environment

We think that, given current software technology, a programming environment suitable for programming education must fulfil the following requirements:

- It must be an integrated programming environment.
- It must include tools that exploit language syntax and semantics. (In our case, this requirement must be adapted to the functional paradigm.)

Let us analyze both requirements in some detail.

2.1 Integrated programming environments

Programming environments contain a set of tools that help the programmer to develop programs: editors, compilers, etc. Usage of an environment is facilitated if it is an *integrated programming environment*. It is difficult to define *tool integration* [6], but it roughly means that tools are members of the environment as a coherent whole. Tool integration can be discussed from different agents' point of view and can be implemented with different mechanisms, but the user is more interested in analyzing tool integration with respect to several criteria:

- Presentation. The user's cognitive load necessary to interact with the tools is minimized. This property is usually achieved with similar user interfaces.
- Data. All the information in the environment is managed as a consistent whole.
- Control. The environment utilities can be used flexibly.
- Process. Tools interact effectively to support a given process. Programming environments do not support large software processes [1], but they must include a set of utilities necessary to carry out mundane tasks, such as handling of files.

Usually, functional programming environments include a compiler or interpreter and some additional utilities (e.g. evaluation tracing). However, they are not integrated environments since they do not provide tools and utilities necessary to support a simple programming process (e.g. an editor and some book-keeping facilities). Moreover, the user interface is commonly very rudimentary.

An educational functional programming environment should provide at least the following elements:

- A consistent and user-friendly user interface.
- A unique vision of programs (e.g. text declarations).
- A set of tools that permit to develop functional programs easily. We can identify some tools: an editor, an interpreter, and utilities for management of files and programs.

2.2 Syntax and semantics based tools

Programming has been recognized to be a task hard enough to deserve the best automated tools. In an ideal situation, the computer would carry out all the tedious and repetitive tasks, and would aid the programmer in those more creative tasks. This scenario only seems to be realistic if programming environments (and therefore programming tools) make an exhaustive use of the syntactic and semantic characteristics of the language.

Moreover, availability of this kind of tools is almost mandatory in educational environments. In effect, a comprehensive computer science education must include and bring together theoretical, experimental and engineering aspects [7]. Programming environments are the tools used for programming experimentation, so they should be designed, whenever possible, to relate theory to practice. In the case of functional programming, a number of experimental tools can be designed on a theoretical base.

2.2.1 Tools based on operational semantics

Functional languages have a very simple operational semantics, where functional expressions are evaluated according to a system of rewriting rules and a strategy of application. Tools devised to help debugging functional programs should be based on this semantics, rather than other *ad hoc* views. In particular, the following utilities can be identified:

- Freedom to choose the *evaluation strategy*. The set of available strategies can be restricted to the two most common: eager and lazy evaluation.

- Flexible use of the operational semantics in tools that *trace program execution*. This facility can be used to both understand program execution and debug programs. The programmer could control program execution so that evaluation advances a different number of rewriting steps. At least three kinds of advance are meaningful:

  a) Advance one step: only the first rewriting step is applied to the expression.
  b) Advance as far as a *break point*: the expression is rewritten until a break point is reached. Functions can be marked as break functions. A break point appears when the next rewriting step is the application of a break function.
  c) Complete the evaluation: the expression is rewritten as much as possible. The final expression is the value of the original expression.

An example can illustrate these ideas further. Suppose we have the following program that determines whether an element is contained in a list:

```
infix or : 4 ;
dec or : truval # truval -> truval ;
--- true or _ <= true ;
--- false or b <= b ;

dec member : alpha # list(alpha) -> truval ;
--- member (_,nil) <= false ;
--- member (x,y::l) <= (x=y) or member (x,l) ;
```

Suppose the programmer wants to evaluate the expression:

```
member (4,[4,5,6]) ;
```

A complete evaluation returns the value of this expression:

```
true : truval
```

However, evaluation can proceed in more detailed steps. Suppose the eager strategy is selected. Step by step evaluation would provide a long sequence of intermediate expressions:

```
member (4,[4,5,6])
↓
(4 = 4) or member (4,[5,6])
↓
true or member (4,[5,6])
↓
true or ((4 = 5) or member (4,[6]))
↓
true or (false or member (4,[6]))
↓
true or (false or ((4 = 6) or member (4,nil)))
↓
true or (false or (false or member (4,nil)))
↓
true or (false or (false or false))
↓
true or (false or false)
↓
true or false
↓
true : truval
```

If *member* is a break function and evaluation proceeds through successive break points, a shorter sequence that only shows recursive applications is obtained:

```
member (4,[4,5,6])
↓
true or member (4,[5,6])
↓
true or (false or member (4,[6]))
↓
true or (false or (false or member (4,nil)))
↓
true : truval
```

On the other hand, the differences between eager and lazy evaluation can be appreciated if lazy evaluation is selected and step by step evaluation is selected:

```
member (4,[4,5,6)
↓
(4 = 4) or member (4,[5,6])
↓
true or member (4,[5,6])
↓
true : truval
```

Notice the great difference between this way of watching expressions evaluation and usual tracing and breaking points facilities. Our proposal shows different but complete expressions during the evaluation, while usual tools only show function applications out of the context where they appear. We have a problem when expressions are very large, because they get unreadable, but we propose a solution in next subsection.

2.2.2 Tools based on syntax and static semantics

Functional languages have a simple syntax and static semantics. This simplicity makes feasible the integration of syntax and static semantics in tools. This can be achieved with several tools:

- *Syntax-directed editors*. Syntax-based editors alleviate the programmer from typing whole programs, but produce these programs under programmer guidance. This kind

of editor is useful when the effort necessary to press the appropriate keys is less than that necessary to type the corresponding text. This effort is worthwhile for expressions and statements with a fixed format, and to generate reserved words, but not for arbitrary expressions. As a consequence, a more appropriate editor for functional programs is a mixed text and syntax directed editor; functional expressions would be typed by the programmer and other parts would be produced via commands.

For instance, let the initial state of the editor be:

```
<declaration>
```

where we want to edit function *member*. Now, a *expand function declaration* option must be selected and the previous declaration will be converted into:

```
dec <function identifier> : <data type> ;
<equation>
```

Now the programmer could write the identifier and the data type of our function, giving rise to:

```
dec member : alpha # list(alpha) -> truval ;
<equation>
```

so that the programmer could choose option *expand equation*, producing:

```
dec member : alpha # list(alpha) -> truval ;
--- member <pattern> <= <functional expression> ;
<equation>
```

The pattern and the functional expression would now be typed by the programmer and the second equation would similarly be expanded in a syntax-directed way. The session would continue until the whole program were developed.

An attractive feature of syntax-directed editors is the possibility of making *static semantic checks*, e.g. whether an identifier has been declared before. In effect, syntax-directed editors can be implemented with an internal representation of the syntax of the program, complemented with static semantic information. As a consequence, semantic checks can be easily accomplished.

The most interesting semantic checks in functional programs are *data type checks*. In particular, functional expressions could be type checked; checks of pattern completeness and consistency could also be included.

For instance, during the previous editing session, the editor should report on a pattern overlapping between the next two equations:

```
--- member (_,nil) <= false ;
--- member (x,nil) <= <functional expression> ;
```

and the user would modify appropriately the second pattern.

- Other syntax-directed tools. Syntax support can be used in other tools apart from editors. For instance, we have seen before that evaluation tools based on operational semantics show clearly the program evaluation process, but the user can get lost when

expressions are very large. One solution is to have the possibility of browsing expressions under syntax support. In this way, a large expression obtained during the evaluation process could be easily browsed using usual facilities in syntax-directed editors, such as *ellipsis*.

The user could even select options for evaluation. For instance, if the last step-by-step evaluation were performed keeping anything but recursive calls under ellipsis, the resulting trace would be:

```
member (4,[4,5,6)
↓
 ... member (4,[5,6])
↓
 ... member (4,[5,6])
↓
true : truval
```

3 HIPE: Hope+ Integrated Programming Environment

An environment prototype based on the previous ideas has been developed. The environment is called HIPE (for *Hope+ Integrated Programming Environment*), in a trial to find a name similar to the name of the language. HIPE includes some of the utilities and tools described in the previous section.

3.1 User interface

The environment is integrated with respect to some of the aspects detailed in section 2.1, but from the user point of view, the most important aspect is presentation integration. In this matter, HIPE has an homogenous user interface. The dialog is based on *text menus* and *text editors*. Menus are used to select operations of the environment (e.g. storing a program on a file or evaluating a given expression), and editors are used to introduce data. There are line editors to introduce strings of characters (e.g. a file name), and screen editors to introduce larger amount of data (e.g. a program or a functional expression).

The use of menus prevents the user from learning a command language to handle the environment. It also prevents her from making certain input mistakes, since the user does not write what she wants, but merely chooses it. The selected menu format is almost standard in many commercial products: hierarchical menu structure, a main menu bar, secondary pulldown menus, option selection with the cursor or with especial keys, etc.

The presentation of the user interface is also homogenous. The screen is divided into three parts: a top line with the menu bar, a bottom line for informative messages, and the rest of the screen as working area. Different colors are used, each one with a different role (e.g. purple is used to report on errors produced on handling the environment).

3.2 Utilities and tools

The main menu consists of four options that classify the utilities and tools of the environment:

- *Handling of files and termination.* The environment has the usual options for interacting with the operating system: change the active unit or directory, consult the contents of the active directory, load the program contained in a file, store a program to a file, and terminate the environment execution. The menu also contains a file

editor facility that can be used to edit text files (e.g. to see whether a program file is appropriate to be loaded or to examine a file which contains an expression evaluation).

- *Editing Hope$^+$ programs.* The user can edit functional programs with a program editor (different from the file editor). For the sake of flexibility, there are two editing modes: the user can edit the whole program or a single declaration (restricted to be a data type or a function declaration). The editor is full screen, like the file editor and the expression editor. When an editing session finishes, the program is analyzed and stored in the environment data structures.

- *Evaluation of functional expressions.* Functional expressions can be edited (again, with an independent editor) and evaluated. For the sake of convenience, the environment keeps a list of expressions, so that the programmer can edit a new expression, or modify or select an existing one. The user can also select the evaluation strategy, and mark and unmark existing functions as break functions. Finally, the selected expression can be evaluated with the active strategy and under user control; evaluation can advance in the three ways identified before: one step, as far as a breaking point, and to the end. Expression evaluations can be saved in files if desired.

- *Book-keeping of the environment.* A set of utilities allows the user to easily handle the declarations included in the environment. Thus, the set of program identifiers can be looked up under their syntactic category; there are five categories: type variable, data type, constructor, function and module use. The declaration of a selected identifier can be browsed or deleted, and the identifier can be renamed. The same operations can also be done (with the exception of deletion) on modules loaded in the environment. Finally, it is possible to delete all the declarations in the environment.

4 Future developments and experience

As the reader may have appreciated, HIPE is a first step towards the fulfilment of the requirements described in section 2. We think that the integration described in subsection 2.1 has been mostly achieved. Likewise, we have developed tools described in subsection 2.2.1, but tools described in subsection 2.2.2 still have to be developed. Our next goal is to develop a mixed editor based on text and syntax, with a semantic base that makes possible static semantic checks.

Something we have not dealt with is our experience with the HIPE environment. Unfortunately, HIPE has not been used by students up till now. We first need to get a high degree of confidence about the quality of the HIPE environment, that is, we need to prove correctness, completeness, robustness and efficiency of the environment utilities and tools. Otherwise, we would risk assigning laboratory problems in functional programming that were unattainable. The available environments are not especially good, but have been proven over many years of use. We plan to use HIPE experimentally during the next school year with a group of students, so that in the subsequent year it can be used with confidence.

References

1. Dart, S.A., Ellison, R.J., Feiler, P.H., Habermann, A.N., Software development environments. Computer 20, 11, 18-28 (1987).
2. Field, A.J., Harrison, P.E.: Functional Programming. Addison-Wesley 1988.

3. Hudak, P.: Conception, evolution and application of functional programming languages. ACM Computing Surveys 21, 3, 354-411 (1989).
4. Perry, N.: Hope+. Technical report IC/FPR/LANG/2.5.1/7, Dept. of Computing, Imperial College, University of London, 1989.
5. Sánchez-Calle, A., Velázquez-Iturbide, J.A.: Fun, rigour and pragmatism in functional programming. SIGCSE Bulletin 23, 3, 11-16 (1991).
6. Thomas, I., Nejmeh, B.A.: Definitions of tool integration for environments. IEEE Software, 9, 2, 29-35 (1992).
7. Tucker, A.B.: Computing Curricula 1991. Communications of the ACM 34, 6, 68-84 (1991).
8. Velázquez-Iturbide, J.A., Belmonte-Artero, M., Castillo-Sánchez, J.A.: Un entorno integrado de programación para el lenguaje funcional Hope+. In: XVIII Conferencia Latinoamericana de Informática (PANEL'92). Proceedings, 1218-1226, Las Palmas de Gran Canaria 1992.
9. Velázquez-Iturbide, J.A., Belmonte-Artero, M., Castillo-Sánchez, J.A.: Arquitectura de un entorno integrado de programación funcional. In: Primer Congreso Nacional de Programación Declarativa (PRODE'92). Proceedings, 345-348, Madrid 1992.

Cognitive Bases for Communication and Learning

Don .G. Bouwhuis
Institute for Perception Research/IPO, Eindhoven, Netherlands

Abstract Teaching and Learning are quite complementary concepts, yet in educational theory far more attention is paid to teaching than to the process of learning. Teaching is a particular form of communication, in which the teaching partner has by default the initiative and controls the knowledge transfer. Communication between humans, however almost always takes the form of a dialogue, in which specific rules of information transfer and turn-taking apply. Central to the notion of a dialogue is the existence of feedback, which indicates the momentary information state of the other partner. While relatively detailed theories of dialogue structures exist, they are rarely applied to teaching situations, where it is of utmost importance to ensure whether the learner has indeed processed the transferred information successfully.

Another issue is that of the many kinds of learning. Learning starts right from birth, but ordinarily very few topics are deemed sufficiently worthwhile to teach explicitly, and to develop educational theories about. Mostly they apply to the learning of complex highly cognitive skills, like language learning, specific mathematical and physical problems, as well as fact learning, such as geography. In order to learn these things, considerable learning must have taken place before to acquire all of this successfully.

There is apparently a strong, autonomous learning capability present in humans, that enables them to take increasingly advantage of environmental variables. There is, unfortunately, very little scientific understanding of this in-built capability to learn, except that it is at least as powerful as the most efficient training schedule ever devised.

Whereas traditional educational theory emphasizes the learning of predominantly declarative and to some extent procedural knowledge, much of human behaviour is essentially based on skills. Unlike declarative knowledge, skills do not develop overnight, are acquired in single trial learning or result from a sudden flash of insight. Skills develop slowly over time and their progress is difficult to track, both for the learner as well as for the teacher.

This makes it hard for any teaching party to infer what the level of skill and knowledge is at any particular moment. Diagnosis of erroneous behaviour is especially hazardous, as cognitive behaviour takes place at so many different levels of abstraction, that correct identification of the presumed error can only be feasible when comprehensive information concerning the interactive instructional situation is available.

These observations indicate strongly that an empirical basis for the design of instructional systems is imperative. This holds on many counts; not only with respect to acceptability for the learner. It holds for the evaluation of the effectivity of the instructional system, the implementation of the didactic measures, and for the design principles. Even then it is unsure how the system will behave in educational practice, and for how long. Recent findings suggest that here too the empirical and social aspects play a determining role.

Activity Theory: Its Implications for Man Machine Communication

Victor Kaptelinin

Psychological Institute, Russian Academy of Education, 9 "V" Mokhovaja Str., 103009 Moscow, Russia

Abstract. Last years there has been growing interest to potential application of activity theory -- the leading psychological approach in former Soviet psychology -- to the problems of human computer interaction (HCI). The paper analyses the reasons why HCI experts are looking for an alternative to the currently dominating cognitive approach. The basic principles of activity theory are presented and discussed within the context of man machine communication. The paper concludes with outlines for potential impact of activity theory on studies and design of man machine communication.

Keywords. Man machine communication, psychology of human computer interaction, activity theory.

1. The current need for a theory of human computer interaction.

It is generally accepted that the lack of an adequate theory of human computer interaction (HCI) is one of the most important reasons why the progress in the field of HCI is relatively modest, comparing to the rate of the technological development. People coming to the field of HCI from different disciplines -- psychology, computer science, graphics design, etc. -- have serious troubles with coordinating and combining their efforts. It can be illustrated by the HCI curricula for undergraduate and graduate students. Usually these curricula present a mixture of pieces of knowledge from various disciplines, not an integrated perspective.

Then, traditional conceptual approaches in HCI cannot provide an appropriate basis for addressing many important aspects of HCI, including Computer Supported Cooperative Work (CSCW) and cross-cultural aspects of computer use. So, it is not surprising that the impact of HCI studies on the current design practice is rather limited. Actually, the user interface design is mainly based on intuition and expensive trials and errors.

Last years there were considerable debates on what should be a suitable HCI theory (see Carroll ea., 1991). Probably the major trend in these debates has been the growing dissatisfaction with the dominating approach in the field, i.e. with cognitive psychology (Bannon, 1991; Wood, 1992; Monk ea., 1993). In contrast to the general agreement on that the current attempts to apply cognitive psychology to studies and practice of HCI are not too successful, there is little agreement on what are the most promising ways of further theoretical development. The answers to this question vary from an enrichment of the traditional cognitive scheme (Barnard, 1991) to a radical shift of the paradigm from scientific experimental studies to the use of ethnographic methodology (Monk ea., 1993).

In this period of theoretical uncertainty (that also means that there is a good opportunity for a new breakthrough in the field) there is growing interest to a psychological tradition that comes from Russian psychology, namely, to the so called "activity theory". This interest was greatly stimulated by Susanne B¬dker's works (1989, 1991). She was the first Western researcher who presented the basic ideas and potential benefits of activity theory to the international HCI community. Recently, a number of papers discussing the activity theory approach to HCI appeared in major international journals and conference proceedings (Bannon, B¬dker, 1991; Draper, 1993; Kaptelinin, 1992a; Kuutti, 1992; Kuutti, Bannon, 1993; Nardi, 1992; Norman, 1991; Raeithel, 1992; Wood, 1992).The aim of the present paper is to summarize the current efforts and the implications of activity theory for the field of man machine communication. The rest of the paper discusses the main differences between activity theory and cognitive psychology, reviews recent attempts to apply activity theory to HCI, and outlines some directions for potential development.

2. From cognitive psychology to contextual analysis of HCI.

According to cognitive psychology, human mind is a special kind of information processing unit. There are various approaches to the architecture of cognition proposed, all of them differentiate between three basic modules, or subsystems, of cognition: (1)sensory input subsystem, (2)central information processing subsystem, and (3)motor output subsystem. Another fundamental idea underlying most cognitive models is the idea of levels of processing. Essentially, this is the dimension of concreteness /abstractness. Input and output represent low levels of human information processing, since they deal with the "raw" data of the external reality. Higher level processing provide identification and classification of the data, their assimilation into mental representations, understanding, analysis, decision making, etc. For a specific action to be made, abstract goals and strategies should be formulated in a concrete form. In other words, the information is processed in both directions: from reality to models and from models to reality.

Concepts and ideas of cognitive psychology have direct analogies in computer science (actually, there are concepts and ideas "imported" to psychology), and the differences in terminology used in these two disciplines are not too large. Apparently it was the major factor determined the dominating role of cognitive psychology in the field of HCI. The subject matter of HCI, from the traditional cognitive point of view, is the system composed of two information processing units -- the human being and the computer, -- so that the output of one unit enters the other's input, and vice versa. In other words, man machine communication can be described as an "information processing loop" (see Figure 1). The advantages of this scheme are rather obvious. First, it provides the coherent description of the whole system of man machine communication within the information processing framework. Second, it structures the problem space of HCI in a useful way. Such aspects of human computer interaction as presentation of the information to the user, user's perceptions, mental models, user's control of the system, input devices, user interface vs. functionality of the system, can be easily located within this scheme.

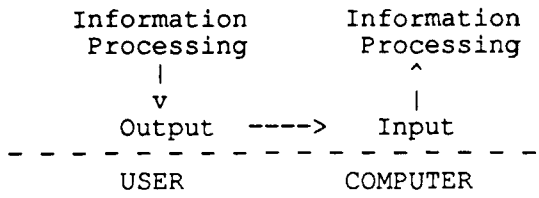


Fig. 1. The "information processing loop" of man machine communication.

The idea of levels of processing has also had great impact on studies of HCI. The hierarchical structure of human computer interface was clearly formulated by Moran (1981). He identifies five levels, namely: the task level, the semantic level, the syntactic level, the level of interaction, and the level of physical devices. This structure is explicitly design - oriented. In a sense, it is supposed to support an analogy of the top-down programming in the domain of the user interface design.

So, it appears that cognitive psychology can be successfully applied to a number of problems of man machine communication. However, this approach has also some serious limitations. First of all, the "ecological validity" of cognitive psychology is questionable. It is remarkable that convincing arguments against ecological validity of cognitive psychology were formulated by the person who gave the name to this approach (Neisser, 1981). Recently, the lack of ecological validity of experimental methods was claimed in a discussion of ethnographic and experimental perspectives in the studies of computer mediated communication (Monk ea., 1993). It was stated that within the experimental framework "an effort is made to wipe out context rather than to understand it" and that "little or no attention is paid to subjects' ideas, thoughts, beliefs about what is being studied" (ibid., p.4).

Second, the information processing loop mentioned above is closed. It is very difficult or even impossible to take into consideration the phenomena that exist outside this loop. It is obvious, however, that man machine communication can only be understood within a wider context. People use computers to create documents, to communicate to other people, etc., i.e., to achieve some goals that are meaningful beyond the situation of computer use. Actually, the "task level", according to the abovementioned hierarchy proposed by Moran, is supposed to put the computer use into the right global context. Yet the relevant concepts and procedures were not articulated, and the models of human computer interaction based on Moran's hierarchy (see Nielsen, 1986; Clark, 1986) are just the models of the closed information processing loop (or a hierarchy of virtual loops).Third, finally, the nature of human mind cannot be completely understood from the information processing point of view alone. Human values, emotions and other important aspects of real life experience can hardly be addressed successfully by cognitive psychology. Interpersonal relations cannot be reduced to information exchange.

Therefore, the theoretical potential of cognitive approach is limited. It does not provide an appropriate conceptual basis for studies of human computer interaction in social, organizational, and cultural context, in relation to goals, plans and values of the user, in the context of development. In other words, the needs of the current studies of HCI, that concentrate not only on the low level events of computer use, but on the higher level events as well (Grudin, 1990), stimulate the search for a theory that provides an appropriate framework for analysis of the context of man machine communication.

There are several candidate approaches, including situated actions approach, distributed cognition, and activity theory (see Nardi, 1992). Next section presents the basic principles of activity theory that determine its conceptual potential for studies of human computer interaction.

3. Basic principles of activity theory.

The general philosophy of activity theory can be characterized as an attempt to integrate three perspectives, namely: 1)the objective one, 2) the ecological one, and 3) the socio-cultural one. Alike cognitive psychology, and unlike some other psychological theories, it tends to be a "real", i.e. a natural science - like theory. Alike J.Piaget's and J.J.Gibson's approaches and unlike traditional cognitive psychology, it analyzes human beings in their natural environment. Besides, activity theory takes into account cultural factors and developmental aspects of human mental life. This brief characteristics can be clarified as follows (see also Asmolov, 1983; B¬dker, 1991, Leontiev, 1978, 1981; Stetsenko, 1989, Wertsch, 1981).

The most fundamental principle of activity theory is the principle of "unity of consciousness and activity". By "consciousness" in this expression is meant human mind as a whole. By "activity" is meant human interaction with the objective reality. This principle states, therefore, that human mind emerges and exists as a special component of human interaction with the environment. Mind is a special "organ" that appears in the process of evolution for to help beings to survive. So, it can be analyzed and understood only within the context of activity.

The next principle is "object-orientedness". This principle specifies the activity theory approach to the nature of environment human beings are interacting with. Unlike theories by Piaget and Gibson, activity theory considers social /cultural properties of environment to be as objective as physical, chemical, or biological ones. Really, these properties exist regardless of our feelings about them. "The object is a book" is no less objective property of a thing than that "the surface of object mostly reflects the light of the red spectrum" (i.e., that the object is "red").

So, human beings live in the environment that is meaningful in itself. This environment consists of entities that combine all kinds of objective features, including the culturally determined ones, which, in turn, determine the ways people act on these entities. The principle of object-orientedness is in obvious contrast with the cognitive approach assumption that human mind contacts reality only through the low level input /output processes.

The third basic principle of activity theory is "hierarchical structure of activity". Activity theory differentiates between processes of various levels (or, rather, groups of levels) taking into consideration the objects these processes are oriented at. Activities are oriented at motives, i.e. the objects that are impelling by themselves. Each motive is the object, material or ideal, that satisfies a need. Actions are the processes subordinated to activities, they are directed at specific conscious goals. The dissociation between objects that motivate human activity and objects this activity is immediately directed at, according to activity theory, is of fundamental significance. Actions are realized through operations that are determined by the actual conditions of activity.

The importance of these distinctions is determined by the ecological attitude of activity theory. In real-life situation it is often necessary to predict human behavior. For this purpose it is of critical importance to differentiate between motives, goals, and conditions. In particular, people behave differently in different situations of frustration. When operations are frustrated (familiar conditions

are changed) people often do not even notice it, automatically adapting themselves to the new situation. When a goal is frustrated, it is necessary to realize what to do next and to set a new goal. But it is often done without any negative emotion. When a motive is frustrated, people are upset and their behavior is most unpredictable.

Therefore, to understand and to predict the changes of people's behavior in different situations, it is necessary to take into account the "status" of the behavior in question, that is, is it oriented to a motive, to a goal, or to actual conditions. That is why activity theory differentiates between activities, actions, and operations. The criteria for separation the hierarchy of processes into these three parts are: 1)is the object the given process is oriented to is impelling in itself or its role is an auxiliary one (differentiates between activities and actions), 2)is the given process automatized or not (differentiates between actions and operations).

Fourth principle of activity theory is the principle of internalization /externalization. This principle describes the mechanisms underlying the originating of mental processes. It declares that mental processes derive from external actions over the course of internalization.

The concept of internalization was also introduced by J.Piaget (1966), but the meaning of this concept within activity theory is somewhat different. According to L.Vygotsky (1956), internalization is social by its very nature. The range of actions that can be performed by a person in cooperation with others composes the so called "zone of proximal development". In other words, the way human beings acquire new abilities can be characterized as "from inter-subjective mental actions to intra-subjective ones". The process opposite to internalization is externalization. Mental processes manifest themselves in external actions performed by a person. So, they can be verified and corrected, if necessary.

Fifth principle is "mediation". Human activity is mediated by a number of tools, either external (like hammer or scissors) or internal (like concepts or heuristics). These tools imply specific ways of operating on them, i.e., the ways of action developed over the history of society. The use of these culture-specific tools shapes the way people act and, through the process of internalization, greatly influences the nature of mental development. So, tools are the carriers of cultural knowledge and social experience. Tool mediation is even more important source of socialization than the formal education.

The mechanism underlying tool mediation is formation of the so called "functional organs". The latter are the systems combining natural human abilities with the capacities of external components -- tools -- to perform a new function or to perform an existing one in more efficient way. For example, human eyes equipped with glasses compose a functional organ that provides better vision.

The last (but not least!) principle is the principle of development. According to activity theory, to understand a phenomenon means to know how it was developed into its existing form. It is the principle of development that gives an opportunity to conduct thorough, scientific analysis of complex phenomena while avoiding mechanistic oversimplifications.

The principles described above are not isolated ideas. They are closely interrelated; the nature of activity theory is manifested in this set of principles taken as an integrated whole.

4. Activity theory and human computer interaction.

According to activity theory, computer is just another tool that mediates interaction of human beings with their environment. The only way to come to an adequate understanding of human computer

interaction is to reconstruct the structure of activity computer use is incorporated into. As it was stated by Kuutti (1992), activity provides "minimal meaningful context" for human computer interaction. The relevant questions, arising when computer use is considered from the point of view of activity theory, are: What is the hierarchical level of human computer interaction within the structure of activity? Does computer use correspond to the level of particular activities (as it sometimes is in "hackers"), or to the level of actions (e.g., when the user is the learner of a system), or to the level of operations (when computer skills are automatized)? What are the tools, other than computer ones, available to the user? What is the structure of social interactions surrounding computer use? What are the objectives of computer use by the user and how are they related to the objectives of other people and group /organization as a whole?

The above questions seem to be too global and loosely related to the practice of user interface study, evaluation, and design. However, when these aspects are not taken into account, some very negative consequences can take place (and regularly occur), including the lack of software usability (Grudin, 1991a, 1991b) and the wrong choice of the software that is not suited to a specific culture (Nakakoj, 1992).

Another general idea directly relevant to the field of human computer interaction is the idea of development. The importance to analyze computer use within the developmental context is relevant to both individual level and group/organizational one. An assimilation of new technologies causes emerging of new tasks (the so called "task-artifact cycle", see Carroll ea., 1991). A possible way to cope with unpredictable structural changes of users' activity is to make the users able to customize the system according to their current needs (Henderson, Kyng, 1991). Yet it is not a universal solution, since users often need substantial assistance even in formulating their own needs. So, a conceptual analysis of the basic factors and regularities of organizational development (i.e., the development of organizational activity) is needed to predict this development and to provide an efficient use of information technologies.

The development of the individual expertise is also an important factor that is not adequately addressed by cognitive approach. The cognitive models of skill acquisition, based on ideas of procedural knowledge compilation or chunking, have troubles with accounting for qualitative changes cognitive skills undergo in the process of their development (see, e.g., Kaptelinin, 1992b). Yet these very transformations can be studied and predicted from the standpoint of the theory by N.Bernstein (1966), the approach that is usually closely associated with activity theory.

The tool mediation perspective implies the structure of human computer interaction that is radically different from the information processing loop. The components of the structure should be not only the user and the computer but also the object the user is operating on through the computer application, and other people the user is communicating with (B¬dker, 1991).

It means, in turn, that there are actually two interfaces that should be considered in any study of computer use: the human /computer interface and the computer /environment one (see Figure 2)

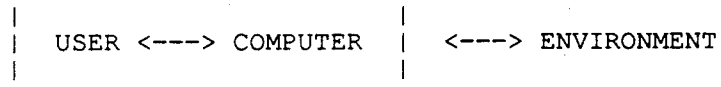


Fig.2. Two interfaces in human computer interaction

So, the traditional interface is not only the border separating two entities but also the link which provides the integration of computer tool into the structure of human activity. The mechanisms underlying this integration can be understood from the activity theory point of view as formation of a functional organ. It means, therefore, that computer applications are the extensions of some natural (pre-computer) human abilities. One of the most important functions of computer tools in the structure of human activity seems to be the extension of the cognitive structure which is referred within the activity oriented approach as the "internal plane of actions" (IPA). The equivalent of IPA within the cognitive tradition can be defined as the mental space where mental models are located. Its function is to simulate potential outcomes of possible events before making actions in reality.

In sum, activity theory provides the wider theoretical basis for studies of man machine comminucation, than cognitive psychology do. It can account for social interactions and for cultural factors, for the developmental aspects and for higher level goals and values. At the same time this conceptual framework does not reject the experimantal results and techniques accumulated within the cognitive tradition. From the point of view of M.Cole, "... US standard cognitive psychology is a reduced subset of a cultural-historical activity approach--without realizing it." (Cole, personal communication, October 1992). Actually, if we compare the information processing loop (see Figure 1) and the tool mediation scheme (see Figure 2), we can see that the former can be easily put into the context of the latter.

5. Prospects for the future development of activity-oriented approach to man machine communication.

One fundamental difficulty related to building up a theory of human computer interaction is the changing nature of the subject matter of the study. In contrast to physical laws, the laws of human computer interaction are not necessarily invariant over time. If the current ways, styles, standards, etc., of computer use are only analyzed, the results of the study are inevitably obsolete soon after (or even long before) they are formulated. Activity theory puts man machine communication into the context of basic, invariant principles underlying human activity, so it provides better chances for creating a theoretical framework that has a predictive potential.

The attempts to apply activity theory to the field of HCI were mainly made during last few years. Most papers, including the present one, are intended just to describe the basics of activity theory and to discuss its general plausibility. However, there are also some cases of the "real" use of activity theory as a conceptual tool in approaching actual problems. These efforts include the analysis of some conceptual problems related to the meaning of the term "interface" (Kuutti, Bannon, 1993), the "mapping" technique that makes it possible to construct a structured two-dimensional representation of the process of computer use and to identify the critical events that take place over this process (B¬dker, 1993), and the development of the "cognitive - cultural" approach to collaborative writing (Wood, 1992).

On my view, there are good reasons to expect more tangible outcomes from activity oriented approach in the coming years. First, I believe new model of human computer interaction will substitute the information processing loop underlying cognitive approach. This model will identify and present in a thorough way the most important aspects of computer use by individuals and groups /organizations. This model will hopefully provide various parties, involved in studies and design of

man machine communication, with a system of references that can make their mutual understanding and cooperation more efficient.

Then, activity theory can greatly influenced the methodology of analysis and evaluation of human computer interaction. The results obtained by B¬dker (1993) can be considered as a first step towards the development of methods providing the opportunity to organize either an appropriate field observation or a laboratory study and to obtain valid and reliable data that would be relevant in real-life contexts.

Finally, activity theory can make an important impact on the development of design support tools. Essentially, the design of new interactive system is the design of a new activity, individual or organizational. However, even the perfect design of an ideal activity does not guarantee the success of a system. The transformation of activity from initial to the target state can be too difficult or even painful. I believe, activity theory can be used as a basis for development a descriptional framework that would help designers to capture the current practice, as well as predictive models of potential activity dynamics. Such conceptual tools would enable designers to make appropriate design solutions, especially at early phases of design.

Given into account the increasing interest to activity theory and the rapid growth of publications during last few years, the above prospects are probably not overoptimistic.

References

Asmolov A.G. (1983) Basic principles of the psychological activity theory. In: A.N. Leontiev and Modern Psychology. Moscow. (in Russian)

Bannon L. (1991) From human factors to human actors: The role of psychology and human computer interaction studies in system design. In: J.Greenbaum, M.Kyng (eds.) Design at Work: Cooperative Design of Computer Systems. Lawrence Erlbaum.

Bannon L., B¬dker S. (1991) Beyond the interface: Encountering artifacts in use. In: J.Carroll (ed.) Designing Interaction: Psychology at the human-computer interface. Cambridge: CUP.

Barnard P. (1991) Bridging between basic theories and the artifacts of human computer interaction. In: J.Carroll (ed.) Designing Interaction: Psychology at the human-computer interface. Cambridge: CUP.

Bernstein N. (1966) Physiology of movements and activity. Moscow. (in Russian)

B¬dker S. (1989) A human activity approach to user interfaces. Human Computer Interaction, v.4.

B¬dker S. (1991) Through the Interface: A Human Activity Approach to User Interface Design. Lawrence Erlbaum.

B¬dker S. (1993) Reframing the human computer interaction from the activity theory point of view. The Journal of Psychology (Psikhologicheski Zhurnal) (in Russian)

Borgman C.L. (1992) Cultural diversity in interface design. SIGCHI Bulletin, October.

Carroll J.M., Kellogg W.A., Rosson M.B. (1991) The task-artifact cycle. In: J.Carroll (ed.) Designing Interaction: Psychology at the human-computer interface. Cambridge: CUP.

Clarke A.A. (1986) A three - level human - computer interface model. International Journal of Man-Machine Studies, v. 24.

Draper S. (1993) Activity theory: the new direction for HCI? International Journal of Man-Machine Studies.

Grudin J. (1990) The computer reaches out: the historical continuity of interface design. In: Proceedings of CHI'90. Seattle, WA.

Grudin J. (1991a) Interactive systems: Bridging the gaps between developers and users. Computer, April 1991.

Grudin J. (1991b) Utility and usability: Research issues and development contexts. In: Proceedings of the 1st International Moscow HCI'91 Workshop. Moscow.

Henderson A., Kyng M. (1991) There's no place like home: Continuing design in use. In: J.Greenbaum, M.Kyng (eds.) Design at Work: Cooperative Design of Computer Systems. Lawrence Erlbaum.

Kaptelinin V. (1992a) Human computer interaction in context: The Activity Theory perspective. In: J.Gornostaev (ed.) Proceedings of EWCHI'92 Conference.

Kaptelinin V.N. (1992b) Can mental models be considered harmful? In: Proceedings of CHI'92. Short talks and posters. Monterey, CA.

Kuutti K. (1992) HCI research debate and Activity Theory perspectives. In: J.Gornostaev (ed.) Proceedings of EWCHI'92 Conference.

Kuutti K., Bannon L. (1993) Searching for unity among diversity. In: INTERCHI'93 Conference Proceedings. Amsterdam.

Leontiev A.N. (1981). Problems of the development of mind. Moscow: Progress.

Leontiev A.N. (1978). Activity. Consciousness. Personality. Englewood Cliffs, NJ: Prentice Hall.

Leontiev A.N. (1979). The psychology of image. Vestnik MGU. Psikhologia, 2. (in Russian)

MacLean A., Young R.M., Belotti V., Moran T. (1991) Questions, options, and ctiteria: Elements of design space analysis. Human - Computer Interaction, v.6.

Monk A., Nardi B., Gilbert N., Mantei M., McCarthy J. (1993) Mixing oil and water? Ethnography versus experimental psychology in the study of computer-mediated communication. In: INTERCHI'93 Conference Proceedings. Amsterdam.

Moran T. (1981) The Command Laguage Grammar: a representation for the user interface of interactive computer systems. International Journal of Man-Machine Studies, v.15, 1981.

Munipov V.M. (1983). The contribution of A.N. Leontiev to the development of engineering psychology and ergonomics. In: A.N. Leontiev and modern psychology. Moscow. (in Russian)

Nakakoji K. (1992) Position paper submitted to CHI'92 Workshop "Cross-cultural Perspectives on Human-Computer Interaction".

Nardi B. (1992) Studying context: A comparison of activity theory, situated action models, and distributed cognition. In: J.Gornostaev (ed.) Proceedings of EWCHI'92 Conference.

Niesser U. (1981). Cognition and reality. Moscow. (in Russian, originally published in 1976)

Nielsen J. (1986) A virtual protocol model for computer - human interaction. International Journal of Man - Machine Studies, v. 24.

Nielsen J. (1990) International user interfaces: An exercise. SIGCHI Bulletin, April.

Norman D. (1988) The Psychology of Everyday Things. N.Y., Basic Books.

Norman D. (1991). Cognitive Artifacts. In: J.Carroll (ed.) Designing Interaction: Psychology at the human-computer interface. Cambridge: CUP.

Piaget (1966) Psychology of intelligence. Moscow. (in Russian)

Raeithel A. (1992). Activity theory as a foundation for design. In: Floyd C. ea. (eds.) Software Development and Reality Construction. Berlin, etc.

Shneiderman B. (1992) Human values and the future of technology. The journal of Psychology (Psykhologicheski Zhurnal), 13(3), (in Russian)

Stetsenko A.P. (1990) On the role of the principle of object-orientedness in activity theory (criticisms "from the outside" and "from the inside") In: Activity approach: Problems and prospects. NII OPP, Moscow, 1990 (in Russian).

Vygotsky L.S. (1956) Selected psychological works. Moscow, (in Russian)

Wertsch J. (1981) The concept of activity in Soviet psychology: an introduction. In: J. Wertsch (ed.).The Concept of Activity in Soviet Psychology. Armonk, NY: M.E.Sharpe

Wood C.C. (1992) A cultural-cognitive approach to collaborative writing. In: Human Computer Interaction: Tasks and Organizations. ECCE 6 Conference Proceedings. Balatonfured, Hungary.

MOTION AS A VARIABLE OF VISUAL COMMUNICATION

Thomas L. Harrington[1] and Peter Bidyuk[2]

[1]Fast Motion Perception, 4715 Mayberry Drive, Reno, Nevada 89509, U.S.A.

[2]Department of Mathematical Analysis, Kiev Polytechnical Institute, 37 Pobedy Avenue, Kiev, Ukraine

1 Transforming the Perceptual Task

Often, information that the human brain processes poorly can be transformed into the world of visual motion where it is processed well. Our technological advancement and edcuation have profited many times from such transformations in other realms. For example, humans have difficulty seeing, judging, and remembering quantities such as temperature, torque and weight. So we have transformed the tasks of perceiving these into tasks where instead we process visual forms, numbers; and assess relative visual location: is the pointer to the left, to the right or directly above the mark on the dial? We are relatively good at perceiving form and relative location.

In addition to facilitating the perception of "invisible" or difficult information, the intensity of information flowing through the interface often can be increased--more information can be perceived in less time. Also, motion can be used effectively to cue visual attention and can serve other operational needs to increase the pace and the flexibility of the educational process.

In the design of interfaces for education faster pacing can be important beyond simply getting the student through the material more quickly. Many systems, such as the brain, change modes completely when certain configurations of their parameters reach critical levels. Speed boats hydroplane, kettles suddenly boil and jugglers can juggle. A slight change will virtually destroy the process. Similarly, the brain has dynamic properties that can often be coaxed into synergy: new information comes in and is assimilated and associated before the electrochemical remnants of the older material can die out, and again, a slight delay can destroy the process--attention crumbles and information that is essential to continuing the process evaporates.

Until recently the educational process has been anchored to verbalization, to the turning of paper pages and to laborious visual search. With the slower computers of old, presenting visual motion would have drained too much processing power from the interface. Now however, computers are fast enough to deal with at least simple kinds of motion. Thus it is feasible to synchronize the interface with higher faster operating modes of the learning system--perception, cognition, memory and motivation.

2 The Perceptual Power Of Motion

Each perceptual facility has its own unique profile of power and weakness. In the case of the visual systems that deal with motion, first there is the ability to process vast amounts of information in parallel. For example, one can immediately apprehend the complex changing patterns etched on fields of wheat or on the ocean by the wind. We easily process the complex motions of individual bees in a swarm, we guide our locomotion using complicated patterns of flowing motion.

In addition to simply carrying large amounts of information, visual motion can serve operational needs of the interface. Motion is a very powerful visual grouping agent that can cement multitudes of diverse elements into perceptible units. The swarm of bees stands out from the forest behind. And elements of perceptual groups can readily be regrouped by motion into new configurations.

Motion is a powerful attractor of attention which can allow quick and flexible cueing of what to perceive next, of what to react to and when to react. It is a purveyor of timing and an excellent teacher and historian of sequences of events and actions.

Visual motion is processed in many cases better in the periphery of vision than are other variables. This often allows redistributions of much of a task's information over a broader visual area, leaving the fovea free to perform the many tasks that only it can do. The periphery of vision can hold rosters of entities or of information that can direct awareness without requiring eye movements.

Motion can convey information about hidden variables such as relative mass that can be readily perceived when translated into motion, for example into a collision of two objects on the screen in which the lighter of the two rebounds more vigorously or is deflected more, or into the erratic wobbling of a ball that is heavy on one side.

Motion can be an aid to motivation in that it is often seductively addictive. Humans and other animals spontaneously watch moving configurations such as screen savers and goldfish, and they may interact almost compulsively with moving targets, as in video games.

3 Transforming Educational Variables into Variables of Visual Motion

As a general principle, information can be transformed from the educational milieu into motions of single elements, into motions of fields of elements, and into modulations of the relative motions of the visual elements that make up the fields.

3.1 *The Educational Setting*

The left-hand column of Fig. 1 shows one possible breakdown of hypothetical material to be taught. Each of these variables can be transformed into the types of visual motion at the right of the figure. The task being taught could be medical: the management of a diabetic or other patient. It might be a skill such as the piloting of an airliner, or the batting of a ball; or it could be a set of concepts from the calculus.

3.1.1 *Configurational*

Items in an educational context often should be grouped into specific configurations to facilitate understanding or memorization. But these groupings may change throughout a process so grouping them simply by "proximity," putting members of each group together on the screen, may not be practical. Situations such as this may occur for example when dealing with groups of symbols or numbers such as in the analysis of variance or in the multiplication of two matrices. Also, often items that must be grouped together perceptually are widely separated in space, for instance in the cockpit of an airliner and ways are needed to make them be seen as grouped.

3.1.2 *Statics*

In nearly every educational setting there are numerous relatively static quantities that must be accounted for or portrayed such as mass or velocity, intensity, amounts or geometrical dimensions.

3.1.3 *Control Information*

Much of education involves learning to control systems. To control any system or to understand is dynamic actions and interactions involves monitoring and correcting aspects of the system; then monitoring the effects of the corrections and correcting again. There is often difficulty in portraying complex networks, states and their feedback loops

3.1.4 *Information about Sequencing*

In many tasks from mathematics to music it is necessary to illustrate sequential operations, for instance in teaching a song, or an operation such as "invert and multiply," the order of differentiation in multivariate calculus, the order that an airliner's gauges should be inspected in; the order of procedures when an engine fails, or in an emergency room situation or in long-term treatment.

3.1.5 *Complex Quantities*

Certain quantities, especially those that depend on a number of other variables such as blood sugar or the predicted attitude of the aircraft, are difficult to perceive in relation to one another and to understand in relation to the overall changing context.

3.1.6 *Complex Arrays*

The states and the dynamic relationships of complicated arrays of simple or of complex entities such as the sets of variables that represent the changing state of a nuclear plant or of air traffic at a busy airport may be nearly impossible to visualize and to learn, track or remember when presented numberically or in static forms of display.

3.1.7 *Hidden Features*

Aspects of certain situations or tasks are not perceivable directly. For example, it may be difficult to assess the mass of an object by simply looking, or to assess the potential effect of some object in a dynamic interaction with something else.

Any educational setting can similarly be broken down and transformed perceptually to the world of motion.

3.2 *The World of Motion*

In the right-hand column of Fig. 1 are some of the perceptual aspects of motion into which the information appearing to the left can be transformed. They have been grouped here as follows.

3.2.1 *Basic Motion.*

The most basic aspects of motion, such as the direction, the motion's location, its velocity and acceleration can carry complex information.

3.2.2 *The Type of Motion*

Information can be represented as different types of motion, for instance, by the type of constraints on it--e.g. circular vs triangular; regular vs irregular; periodic; idiosyncratic.

3.2.3 *Motion of Visual Elements in Concert and in Depth*

If visual elements move together as fields, "common fate" will hold them together perceptually. If their individual motions vary some then they may be seen as moving in depth rather than simply moving over the flat display. Information from the educational setting can be transformed into this perceived depth.

3.2.4 *Motion in Time*

The timings of individual motions in a display can bear information. Two or more elements moving separately have dimensions of relative motion that and are accompanied by perceptual effects that neither possesses alone.

3.2.5 *Personality of Motion*

Many kinds of information, such as urgency, can be transformed into a motion's "personality" essentially as any set of characteristics that can be applied to living organisms: energetic, calm, gentle, and so on.

3.2.6 *Natural Motion*

Finally, the visual system is able to process some very specific kinds of information especially well. One striking example is the movement of people. Information about human activity can often be conveyed with minimal stimulation; for example, the perceptual nuances of two people dancing can be shown quite precisely by small lights attached to a few of the joints, viewed in the dark. Accordingly, much information from the educational setting can be represented to perception as dynamic naturalistic dot characatures.

4 Examples

It is possible to transform any of the kinds of information related to the educational setting, shown in the left of Fig. 1, into any of the types and nuances of motion appearing on the right. A subset of these transformations, the ones related to grouping, will be illustrated. It will be left as an exercise to finish transforming each member of the column on the left to each aspect of motion on the right.

4.1 *Using Basic-Parameters-of-Motion to Create Perceived Groups*

If the members of one group of elements, for example icons, points on a display, or cells in a vector or matrix, move and other members move differently, or are stationary, perceptual grouping will usually occur which labels or identifies the members of the group, even though they may be widely separated spatially. Perception of the shape of the group of elements is also generated.

For example, in a dynamic two-dimensional scatter plot, sub-groups will be seen if selected elements simply have different velocities, directions of motion or accelerate differently.

4.2 *Using Type-of-Motion for Perceptual Labelling*

In complex grouping problems it is possible to indicate overlapping or nested groups, for example in a medical school to show epidemidiology as related to a number of variables, or to portray risks to health or intellect in aging or diabetes.

Imagine a complex two-dimensional scatter plot of age vs some aspect of life history such as amount of smoking. Type-of-motion can provide sub-grouping. First, suppose it is desired to indicate gender. Points representing scores of females oscillate sinusoidally, those of males oscillate with a saw-tooth function.

Next, the back-and-forth motions indicating gender are modulated with fast oval motions, as in penmanship's "English ovals" to indicate those who drink alcohol; a nervous jitter superimposed on the motion signifies those who use caffeine, and so forth.

4.3 *Grouping in Three-dimensional Fields of Motion*

Certain types of moving two-dimensional patterns produce perceived motion in depth. Groups of elements, such as data points or icons, can be forced onto different perceived planes of motion, or into clouds, or into arbitrary complex shapes in the third dimension. Also, such forms and clusters can be discovered in data by producing motion in depth that segregates groups that had appeared unitary. Adding a new dimension of space to a display "unsticks" local areas that may previously have been occluded or have appeared to belong grouped with elements that can be seen to be in fact in front of or behind them. There are several forms of three-dimensional data structures.

4.3.1 *Data on Three-dimensional Planes*

Movement can be used to "stick" data points or other elements onto

two-dimensional surfaces that elicit perceptions of three dimensional planes by projecting the three-dimensional scenes mathematically onto the display. For instance, separate groups of points can be pasted in this way to rotating intersecting "pizza" charts where they will be seen as grouped together on their rotating charts and segregated from the others..

4.3.2 *Tangles*

A three dimensional trace it can be viewed more advantageously by creating a kinetic depth effect. This is accomplished by simply projecting a two-dimensional image of a rotating third-dimensional structure onto the screen mathematically. Powerful impressions of depth and excellent perceptions of the structure's form are generated.

To illustrate in the case of loci representing data or theory in space, suppose that the shadow of a tangle of string is projected onto a screen motionless. The tangle, of course, will appear flat, and the intersections of shadows appear to be real intersections. When the shadow is rotated to even the slightest degree the intersections un-group and each section of string can be seen to be behind or in front of the others.

Also, which parts of a shadow belong to one section string and which parts to another becomes quite evident, even though many feet of string are "hopelessly" intertwined.

In the study and description of chaotic behavior the real shapes of strange attractors can be unmasked in this way, phase portraits can take on new meaning.

4.3.3 *Clouds*

In dealing with "clouds" of data, for example 3-dimensional scatter plots, sub-clouds may emerge, and common trajectories of certain groups in space-time may become evident through kinetic depth. For instance, "before-next" plots of data from complex systems can profit from being extended into higher-dimensioned space.

A classic example of these comes from the dripping faucet, or analagous processes such as musical notes as they are played, or EEG or EKG. Data points are plotted as follows: The time between drip one and drip two serves as the "X" coordinate; the time between drip two and drip three as the "Y," for the first point. Then the time between drip two and drip three becomes the "X" coordinate for the next point and the time between drip three and drip four is the "Y" coordinate. This process continues until the desired plot is generated.

Sometimes, however, it is useful to analyze such processes in three or more dimensions. Then a "Z" coordinate is also used: the first time it is the time between drips three and four, then the time between four and five, and so on. Analyzing such plots is much more effective if they are rotated as they are mathematically projected onto the display.

4.4 *Grouping from Natural Stimuli*

Some transformations of information to motion in depth are especially compelling in their production of visual groups because they appear to tap basic

mechanisms of perception that are very specialized for specific natural tasks, such as guiding locomotion or monitoring the movements of another person. Following are two examples employing natural groupings.

4.4.1 *Modulating Perceptual Mechanisms of Surface Formation*

Suppose that that it is necessary to monitor and analyze three related variables and all of their interactions: Blood sugar, level of insulin and blood pressure.

These can be mathematically woven into some natural visual entity--here a moving visual surface, in such a way that their values and their interactions will warp the surface. To do this we transform the variables respectively to three of the variables that dictate the sizes, locations, velocities, and accelerations of the textural elements that make up the surface. These three variables are:

Divergence. As textural elements in the world, such as those on floors or ceilings, move toward us, or we move toward them, they diverge apart optically, as the rails of a railroad do. If divergence lessens then the surface that the elements are seen to lie on appears to tilt, becoming more perpendicular to the line of sight.

Size-change. As we approach textural elements on a surface they become optically larger.

Velocity-change. As we approach and pass objects their angular velocities increase--objects on the horizong barely seem to move, then when abreast the go past quickly.

These three variables working in normal synchrony preserve the shapes of objects we move among, floors, ceilings, etc. and we see a normal world. We are very finely tuned to any aberrations in this world. When their complex natural relationships are disrupted the shapes the brain derives from them are perceptually disrupted too.

Not only changes in the individual variables, but also changes in the complex forms of their mutual high-order interactions become immediately visible. When a visual array such as this is set in motion and warped by changing one of the parameters the perceptual impact is as strong as if one were witnessing a travelling wave in the side of a building or in the pavement. Perceptual leverage is high in that small changes of the variables can cause large changes of the percepts!

4.4.2 *Motion-produced Grouping in Natural Personal Motion*

Motion can group perceptually the separate parts of moving people and animals. A number of investigators have studied the perception of small lights that move independently. One demonstration of amazing perceptual impact, in terms of conveying large amounts of information with simple stimuli, is that in which lights are attached near some of the joints of people who then move in the dark. Even though there is nothing obvious about the physical motions of the lights that would necessarily predict their being grouped one way or another as they move, and even though their paths may be entangled in various ways that would seem guaranteed to preclude any possibility of grouping, "people" in motion are clearly seen. The actors in the dark may dance, perform most actions a body can perform, yet not

only does the brain keep the lights perceptually grouped together somehow, the specific actions of the people themselves are directly perceptible in great detail. For example, the weight of an object that is being lifted in the dark can be assessed fairly well by an observer only observing the motions of these few lights.

Displays that have been stripped of most of their non-essential information, as in kinematic studies and portrayals of performance in sports often end up as displays of moving points because we are so sensitive to them. The grouping is not only completely adequate for holding a changing percept together, it allows more easy comparison between groups as in trying to assess or narrow the differences in motions between an expert in some motor skill and an amateur.

4.5 *Motion, Stereopsis and Self-stereopsis in Grouping*

Most of the above methods can employ stereopsis to extend their usefulness. However, usually there are additional demands for viewing equipment, but a number of very sophisticated devices have been developed. Eventually the specialized goggles that are being refined presently for producing virtual realities and for replacing the mammoth displays of flight simulators will be available for stereoscopic presentation, so it is sensible to assume that any interface that is designed may soon be afforded stereoscopy.

4.5.1 *Self stereopsis*

One method exists that requires none of the new hardware. Fig. 2 shows such a method. It is a "recursive stereogram" (Harrington and Quon, 1989). In order to view the depth in the static version on the page simply cross or diverge the eyes until the two "Q"s with the dots in them merge into three. Depending on the size of the display and other factors the images may fuse if one simply looks at one's own reflection in a screen bearing this display for a few moments, thereby diverging the eyes.

Groups of data can be caused by motion in stereopsis here in a rather different way. For example, it is possible to portray two surfaces in motion, maintaining their integrities even as they cut through one another, by xeroxing Fig. 2 onto a transparency, placing that over the original, and moving it while fusing as directed above. The percepts of two transparent surfaces emerge that are penetrating one another and sliding through one another. Similarly perceptual groupings into surfaces whose parameters can carry transformed information, can also be formed by "flowing" the elements over the surfaces (even though the elements are the surfaces) or by shifting the transformations that produce the surfaces (the latter appearing as cats walking about under a blanket).

In the design of a visual interface, the perceptual skills of the users are important. Some perceptual skill is required for viewing the effects above, and some individuals can never see them. On the other hand, there are certain groups whose members usually can. These are vision scientists, of course, geographers, geologists and radiologists (See Harrington & Quon, 1989).

4.5.2 *"Windshield-Wiper" Stereoptic Phi Movement*

At this point an operational suggestion involving stereopsis to observe phi motion in the absence of a tachistoscope may be useful (personal observations). If the separate frames of a two-frame animation are placed side by side and visually fused as in "cross-eyed stereopsis" and then the eyes are alternately opened and shut or alternately occluded by moving the hand back and forth, the motion, with some practice, can be observed.

4.6 *Motion in Relation to Time as a Grouping Force*

Perceptual grouping can be elicited by the timing with which a motion starts relative to other motions and, for example, how rapidly the motion repeats in time.

4.6.1 *Phase*

Phases of two or several motions can be very effective in discovering or changing, or modulating, a perceptual grouping, or loading it with additional information.

For example, if the variables: blood sugar, pulse rate and blood pressure have been mapped onto divergence, size change and velocity change of a field of moving visual elements, as was discussed previously, and if these variables have phase shifts in their interrelations, this might ordinarily obscure their inter correlations. However, when they are integrated visually onto a moving surface and there are phase differences, distortions in the surface stand out clearly. If the situation is complicated with intricate changes in the phases then complexes of travelling waves may be seen moving about on the surface--again something that the visual system is exquisite at analyzing, just as it perceives every nuance of an incoming wave on the ocean.

Essentially, a visual analysis is possible after this mapping in which regular features, interactions, and non-linear irregularities stand out prominently and instantly across the entire domain of interaction of the variables.

4.6.2 *Timing of Onset*

Timings of the respective motions including appearances and disappearances can be used to modulate the ways that visual elements are perceptually grouped in time. For instance, in a cloud of data points, groups of points can be made to appear and disappear successively, perhaps with some overlap in time to allow the intersections of these subgroups to be viewed. Similarly any local signature such as a cyclic jittering of successive groups can capitalize on timings of these motions to cause the groupings.

4.7 *Perceptual Grouping of Personalities of Motion*

Just as it is impossible to draw a "stick face," a line-like mouth and two eye-like dots that do not have the appearance of some strong personality, it is impossible to portray a motion that does not also have a personality. This is an emergent quality supplied by the brain, with very powerful perceptual leverages. The slightest change on the physical side, perhaps in the corner of a line-like mouth can cause great changes in the percept.

Complex motions, even some simple ones, are seen as energetic, portending doom, happy, constrained or liberated. In general it is possible to emulate most personal qualities a living creature might be endowed with as personalities of motion, and these can cause grouping..

5. Conclusion

Hopefully these examples provide the beginnings of a menu of transformations where motion can be used to convey other kinds of information that is less effective in entraining the brain's mechanisms. In a similar fashion the other educational variables of Fig. 1 can be transformed into any of the variables of motion. The best perceptual re-mapping of course depends on the situation, on the medium, and other aspects of the specific context. Which mapping is chosen will depend primarily on the intuition of the designer, based on preliminary experimentation, hardware, esthetic considerations and on the many trade offs that every educational setting has.

TRANSFORMING EDUCATIONAL INFORMATION TO THE DOMAIN OF MOTION

FEATURES OF THE EDUCATIONAL MATERIAL	THE WORLD OF MOTION
Configuration	The basic parameters
Static quantities	Type of motion
Control information	Fields of motion
Sequencing	Timing of motion
Complex quantities	Personality of motion
Complex arrays	Naturalistic motion
Hidden quantities	

Fig. 1

SELF- STEREOGRAM

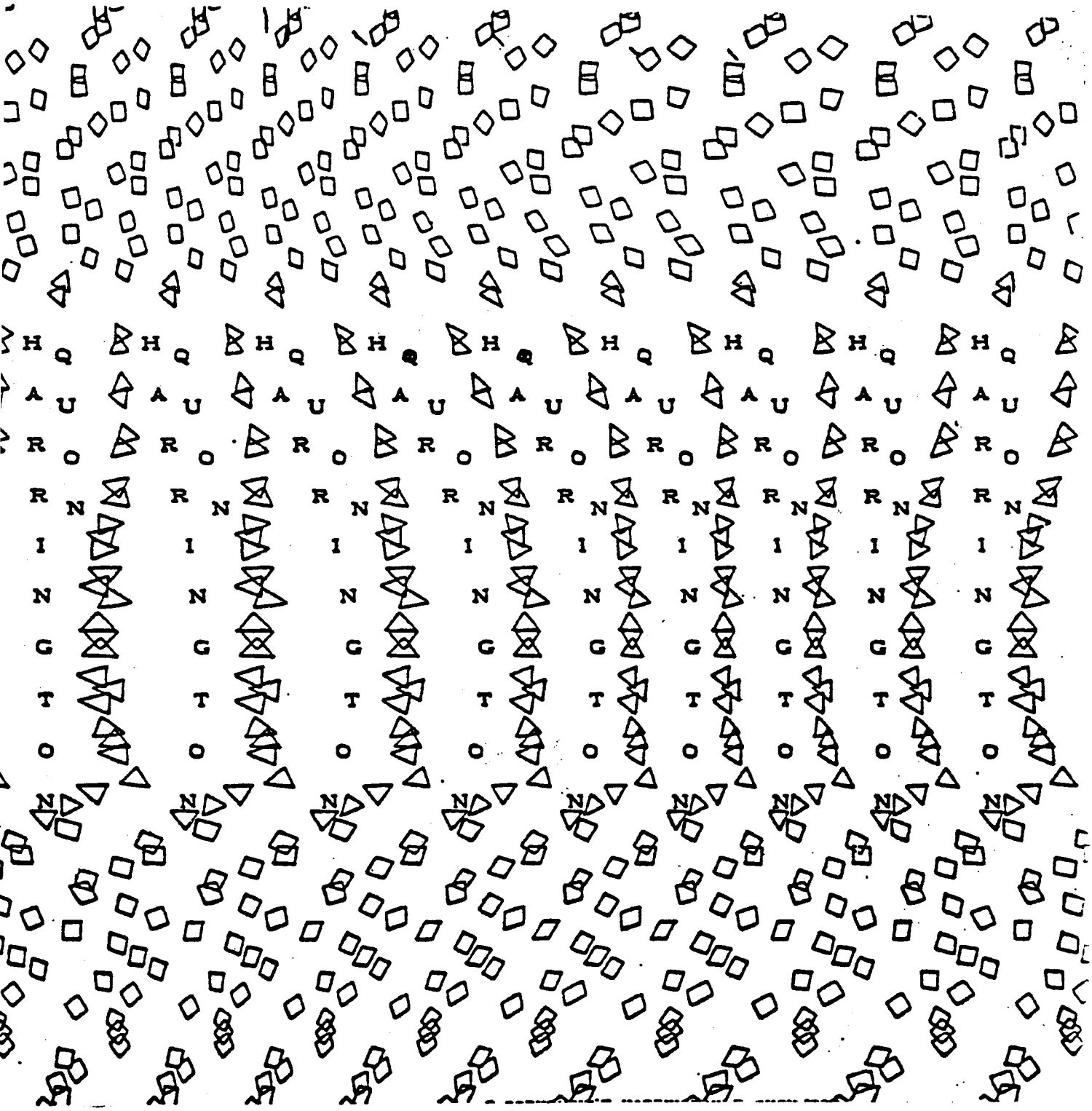


Fig. 2

Cognitive load and the acquisition of a problem-solving skill

R. Van Hoe

Institute for Perception Research/IPO, PO Box 513, 5600 MB Eindhoven, The Netherlands

Abstract

Current theories of learning consider the restructuring of the components of a weak problem solving sequence into a domain-specific procedure as the fundamental learning mechanism in complex knowledge domains. Within the context of cognitive load theory, there is growing evidence that applying weak methods such as means-ends analysis, do not necessarily result in learning, but can in fact induce a high mental load which prevents the problem solver of inducing domain-specific rules. The major goal of this article is to provide a direct test of the cognitive load theory. In two experiments evidence was found that the cognitive load imposed by a weak problem solving activity can retard the acquisition of a problem-solving skill. However, the experiments also illustrate that the major drawback of the cognitive load theory is the central concept of the theory itself. The term "cognitive load" is a rather vague and abstract concept which stands for a set of variables (attention, effort, storage capacity) which can constrain the interaction between processes, and because the cognitive load imposed by a process is not directly measurable, it is not always clear whether a variable or which variable constrains the relation between problem solving and learning.